

System Administration and Resources Management A Case Study of MAMWE (Comoros)

By

Said Abdou Mfoihaya Student ID 113403	Mohamed Ahamada Sohibou Student ID 113425	Ibrahim Ali Algabal Student ID 113407
---	---	---

Supervised by

Faisal Ahmed

Lecturer, Department of Computer Science and Engineering

Tareque Mohmud Chowdhury

Assistant Professor, Department of Computer Science and Engineering

A Thesis submitted to the Department of Computer Science and Engineering (CSE)
in Partial Fulfillment of the Requirements for the Award of the Degree of Bachelor
of Science in Technical Education with Specialization in Computer Science and
Engineering (BSc.TE. CSE)

Department of Computer Science and Engineering (CSE)

Islamic University of Technology (IUT)
Organization of the Islamic Cooperation (OIC)

Gazipur 1704, Bangladesh

September, 2012

CERTIFICATE OF SUCCESSFUL COMPLETION OF THE THESIS/PROJECT

All thanks to the Almighty Allah for his mercy and guidance on us. Within the academic year 2011-2012 we have successfully completed our thesis work on **System Administration and Resources Management A case study of MAMWE**, the Comorian public company providing water and electricity services to the Comorians. This project is undertaken by students of Islamic University of Technology a subsidiary organ of the Organization of the Islamic Cooperation under the supervision of **Tareque Mohmud Chowdhury** assistant professor, and **Faisal Ahmed**, lecturer, department of Computer Science and Engineering. We would like to note that neither this report nor any part of it was submitted anywhere for the award of any degree or diploma. This report is submitted as a partial fulfillment for the award of bachelor degree in technical education with specialization in Computer Science and Engineering.

Dissertation by

Said Abdou Mfoihaya
Student No 113403

Ibrahim Ali Algabal
Student No 113407

Mohamed Ahamada Sohibou
Student No 113425

Supervised by

Faisal Ahmed
Lecturer, Department of Computer Science and Engineering,

And

Tareque Mohmud Chowdhury
Assistant Professor, Department of Computer Science and Engineering

Date/...../.....

Abstract

The issue of administration has been one of the most critical problems that many companies ranging from small, medium, and large companies suffered from in world wide since many years ago. But more recently during the last 30 years with the advent in computer science and information technology, companies are being transformed from the earlier record keeping and information processing based on office hard documents to the latest information processing using computer software. Many companies are based on locally installed systems which cannot overcome the communication problem for companies which are distributed in large geographical area. These systems do not allow their customers to stay in touch with them which was a big challenge for all companies. In this case a more flexible computer system is needed. Today the web offers a very fast way to make communication happen more efficiently from all over the world by using the so called web applications over the internet. Web applications enable companies to setup any number of centers in different geographical areas while being working together in the same software system and to bring customers closer to their services. The aim of this project/thesis is to develop a web application to be used as a standard of administration system and project management tool for a company namely MAMWE which stands for Madji-na-Mwenje ya Komori, a public company in COMOROS providing electricity and water facilities to the Comorians. The company consists of three main head offices, centers, power stations, water stations, substations, and a large number of customers all inside the COMOROS ISLANDS.

Acknowledgement

First and foremost we offer our sincerest gratitude and thanks to the Almighty Allah who gives us the abilities to successfully perform this task. And no doubt without the help of the Almighty Allah this report could not have been completed or written.

It would not have been possible to write this bachelor thesis without the help and support of the kind people around us, to only some of whom it is possible to give particular mention here.

Above all, we would like to acknowledge the advice and guidance of our principal supervisor **Tareque Mohmud Chowdhury**, assistant professor department of computer science and engineering (CSE), Islamic University of Technology (IUT) Organization of the Islamic Cooperation (OIC). This thesis report would not have been possible without his help, support and patience.

We would like also to thank our Co-supervisor **Faisal Ahmed**, lecturer, department of computer science and engineering(CSE), Islamic University of Technology (IUT) Organization of the Islamic Cooperation (OIC), who provides us with the appropriate tools and materials.

We would like also to acknowledge the support and efforts of **Prof. Dr. M. A. Mottalib**, head of the department of computer science and engineering(CSE), Islamic University of Technology(IUT) Organization of the Islamic Cooperation(OIC).

Last, but no means least, we thank our beloved parents who worked a lot from our childhood till today for our lives and success.

For any errors or inadequacies that may remain in this work, of course, the responsibility is entirely our own.

Table Of Contents

CERTIFICATE OF SUCCESSFUL COMPLETION OF THE THESIS/PROJECT	I
Abstract.....	II
Acknowledgement	III
List of figures.....	VII
List of Tables.....	IX
Chapter 1 Introduction	1
1. Overview	1
2. Motivation.....	1
3. Problem Statement.....	2
4. Development Challenges	2
5. Report Outline.....	3
Chapter 2 Literature Review	4
1. Introduction	4
2. Web development strategies.....	4
2.1. Advantages of web application.....	4
2.2. Strategies in web development	5
3. System development life cycle	12
3.1 SDLC Phases	12
3.2 Object-oriented Analysis and Design	13
4. Tools and Technologies.....	14
Chapter 3 System Analysis and Design	15
1. Introduction	15
2. Requirements Definition.....	15
2.1 What.....	15
2.2 Why	21
2.3 Who.....	22
2.4 When.....	23

2.5	How	24
3.	Object Oriented Analysis and Design.....	25
3.1	UML Diagrams for Interactive System	26
3.2	Documenting Use case Diagrams.....	33
4.	Conclusion.....	35
Chapter 4 System Development and Implementation		36
1.	Introduction	36
2.	GUI Design Principles	36
2.1	Understand the people	36
2.2	Be Careful of Different Perspectives.....	36
2.3	Design for Clarity.....	37
2.4	Design Consistency	38
2.5	Provide Visual Feedback	38
3.	Applying Design Principles	38
3.1	Home Page interface.....	38
3.2	Default Home page	39
3.3	Login Window	40
3.4	Management Profile Interface.....	41
3.5	Manager User Guide	41
4.	Initializing and Configuring the System.....	42
4.1	Mamwe Organ Registration	42
4.2	Creating Employee Account Record.....	43
4.3	Resources Registration and Management.....	44
4.4	Generating Salaries	45
4.5	Creating Customer Billing Record	47
4.6	Setting up Employment Information System.....	47
4.7	Creating a Finance Account Type.....	48
4.8	Creating a Revenue Account Type	48
4.9	Assigning Expenses and Revenues to Accounts.....	49
5.	Building Reports.....	50
5.1	Building a Salary Payment Report.....	50
5.2	Building a Service Report	51

5.3	Building Bill Payment report	52
5.4	Report Printing Options	52
5.4	Printed Report Example	53
Chapter 5	Customer Support and Feedback.....	54
1.	Overview	54
2.	Post Implementation Review	54
2.1	When to Review	54
2.2	What to Review.....	54
2.3	How to Review	56
3.	Supports Activities	57
3.1	User training and assistance	57
3.2	Information centers	57
4.	System Maintenance	58
5.	Conclusion.....	58
Chapter 6	Conclusion and Future Works Strategies	59
1.	Overview	59
2.	Conclusion.....	59
3.	Suggestions for future works	60
3.1	Hybrid Profile	60
3.2	Online Service	60
3.3	Remote Voltage Monitoring	60
References	61

List of figures

Figure 2.1 : Client Server Scripting Mechanism	8
Figure 3.1: Levels of Software Requirements	16
Figure 3.2: User Requirements Versus Organizational Structure	16
Figure 3.3: Organizational Levels of Management	17
Figure 3.4: Users Organization Structure for Mamwe	20
Figure 3.5: Kano Model for Quality Requirements	22
Figure 3.6: Requirements Engineering & Management	24
Figure 3.7: Accounting Module Class Diagram	27
Figure 3.8: Accounting Module Object Diagram.....	28
Figure 3.9: Setting Up Account Use Case Diagram	29
Figure 3.10: Expenses and Revenues to Account Assignment Process	30
Figure 3.11: Adding Expense/Revenue payment Use Case Diagram	31
Figure 3.12: Budget Allocation Use Case Diagram.....	31
Figure 3.13: Customer Billing Processing Use Case Diagram	32
Figure 4.1: User Session Life Time	38
Figure 4.2: Default Application Home Page	39
Figure 4.3: Home Page With Login Window Tab Activated	40
Figure 4.4: Task Processing Life Time	41
Figure 4.5: Higher Level Management Profile with default user tasks guide	41
Figure 4.6: Mamwe Organs Registration Form	42
Figure 4.7: Employee detail and Accounts Registration	43
Figure 4.8: Resources registration & Management Service type Checked.....	44
Figure 4.9: Resources Registration & management Project type checked.....	44
Figure 4.10: Resources registration & management Salary type checked with default generator.....	45
Figure 4.11: Resources registration & management Salary type checked - Generating all salaries	45
Figure 4.12: Resources registration & management Salary type checked - Generating a specific salary..	46
Figure 4.13: Resources registration & management Salary type checked - Generating Job based Salaries	46
Figure 4.14: Resources registration & management Customer Billing Creator.....	47
Figure 4.15: Setting up employment related information.....	47
Figure 4.16: Finance Accounts Creation Process	48
Figure 4.17: Revenue Accounts Creation Process.....	48
Figure 4.18: Finance & Revenue to Accounts Assignment Process	49
Figure 4.19: Adding New Salary Payment & Report Making	50
Figure 4.20: Registering service payment & report making	51

Figure 4.21: Customer Billing Confirmation & Report Making 52
Figure 4.22: Customer Invoice Print Preview..... 52
Figure 4.23: Customer Billing Printed Invoice Sample 53

List of Tables

Table 2-1: Availability Percentage to Time Conversion 11

Table 3-1: Mamwe Higher Level Management Activities 18

Table 3-2: Mamwe Middle Level Management Activities 19

Table 3-3: Mamwe Lower Level Management Activities 20

Table 3-4: Setting Up Account Use Case Documentation 33

Table 3-5: Adding Expense/Revenue Payment Use Case Documentation 34

Table 3-6: Budget Allocation Use Case Documentation 34

Table 3-7: Bill Payment Processing 35

Table 4-1: List of reserved Icons and Messages 37

Table 4-2: List of Reserved Words 37

1. Overview

Many years ago, many companies (private and public) in the developing and least developed countries were suffering due to several problems such as administrative and resources managements. One of the most important reason is how to determine an appropriate standard system to which the company is based to conduct their overall activities. Another reason may be the lack of tools to automate, record, and control the company's activities. The third reason we are mentioning here may be the lack of trained manpower. Consequently they are using systems and tools designed for or used by other companies in the developed countries which are not convenient for their actual needs.

This report represents a special case of MAMWE. Throughout the work we should be able to find a convenient solution on the administration and resources management problems of MAMWE. Arriving to a solution is not quite easy specially in today's competing age. It requires a definite range of steps through which the system can be analyzed, designed, and implemented. This chapter will take you through a deep understanding of the system and the steps that we are going to follow in order to achieve our objectives.

2. Motivation

This work is conducted in Islamic University of Technology, a subsidiary organ of the Organization of the Islamic Cooperation. The reason to conduct such a work is to promote our learning in software development. But behind this reason let discuss about our interests in this study.

First of all, developing such kind of system is a key success in our studies. We aim to apply the methods and techniques of the software development process in a real life environment including customer specifications and acceptance. This is in fact an important stage where we will increase our experience in the above stated domain. The result of this work may greatly contribute in the development of MAMWE in different domains. It may serve as a basis for the overall administration and management of the company, a communication system between the company and their customers, and a base for future developments.

Furthermore, the experience of implementing this software system advances in an increasingly resource management environment for MAMWE, in some instances approaching the control of all the company's activities.

Finally, it is in our dreams to develop our communities through the usage of the new technologies. As we may know today's world is referred to as the digital age where software is the key for its development.

3. Problem Statement

The MAMWE company is facing lot of problems in their management system for a very long time. Till today there is not any appropriate solutions for these problems causing a huge range of difficulties in the company. Consequently they are getting enormous losses both within the company as well as in the outside world. Problems range from data storage system, access limitation, poor data manipulation and processing, to the reports building. In this situation the company continues to fall down. Also these problems become the origin of management complexity and corruption.

In our study we assumed that the above stated problems are still existing. However, our recommendation is based on developing a fresh software system to achieve the following goals:

- Creation of a communication system between the company and the outside world
- Facilitating the overall management of the company
- Improving the marketing system of the company
- Creating a distributed working network
- Offer an online presence of the company
- Overcome time consuming and resources wastage problems
- Implementing security issues
- Adapt to new technologies

4. Development Challenges

Several attempts to the development of a new tool for this company have almost all failed from which are tools being used in other similar companies in Africa. But it is clear that copying a system is not a mean for solving one's problem even though both parties share some common properties.

First, for developing a software, there is a very important consideration on the users of the software. When for example we developed a student management system for Islamic University of Technology, the same system may be useless for another school or college or even

for a similar university. The functions implemented in the software have a close relationship on the current situation, the requirements of the users, as well as the current technologies in use.

Second, developing a total software system is not enough. Once a software is complete, other requirements may be needed leading to the development of highly appreciated software. But all the attempts to solve the MAMWE problems came from abroad which causes difficulties in requirements changes.

Third, after the development of the software, this needs to be maintained as many software developers do. But it becomes very difficult as detecting bugs in any software system needs a trained person in the field of software engineering. Basically this task is done by the developers of the software. And as the bugs can't be detected, the whole system is not stable. Consequently the same problems that the company faced previously are more likely to occur again and again.

Finally, the company is wasting a huge amount of money day by day to find a solution for their problems. Unfortunately, they could not reach to the desired solution due to the above stated limitations. We have undertaken this work with the hope of achieving the company's goals through a very flexible software system.

5. Report Outline

Throughout this report, we will discuss as much as possible according to the following topics: First of all we will go through a ***literature review*** where the basic theories behind this project will be discussed. Second is the ***System analysis and design*** whose aim is to analyze the current situation and design a new system based on the available resources. Third, we will go through the application of new design referred as ***System development and implementation***. Fourth, our discussion will be based on the software post implementation or Customer support and feedback. And finally we will close our discussion by summarizing our result and giving an extension of the system for future developments. This last chapter is referred to ***Conclusion and future works***.

1. Introduction

Choosing a common project or theme is quite easy, but making a decision to go with exact and sensible development and research requires great consideration and thoughts while suitable and applicable development and investigation techniques are required with accessible resources.

2. Web development strategies

A Web design strategy pays off for us and our organizations. Whether our site is up and running or we are composing the blueprint for a new one, having a design strategy helps us make sure our site tells our story. The key reason we need one is because this is how we “stick the landing.” It creates a vehicle for our team to capture and document conversations that always seem to go in circles. Once we have it down on paper, we can be assured that everyone is on the same page when we implement the plan and manage the operations. It’s a living document and may change as our site evolves, but at its core, the design strategy tells the story of our site to our team and ultimately our audience.

2.1. Advantages of web application

- The greatest benefit of web application is the 24 hrs & 7 days online availability as well as visibility from anywhere around the world so as to enable the users to draw information at any point of time.
- It is a great way of supporting the personal recommendation of clients.
- It helps your company to increase the product knowledge, sustain communication between you & your prospective customers.
- It increases the popularity of your company.
- It improves the visibility as well as credibility among the clientele.
- It provides all the information regarding the products, the promotional as well as developmental activities.

2.2. Strategies in web development

2.2.1 *Setting up the goals*

Nowadays the web becomes the most popular and useful tool for communicating people all around the globe. The web provides different mechanisms of reducing the costs for the daily life of the management of millions of organizations in the world. It also improves the way an organization makes money and handles its customers via the internet. Let now consider the case of the MA-MWE company. Actually the company provides their facilities to several hundred thousands of customers in the Comoro Islands. Creating a web based application for the company will overcome many problems that both the company itself and their customers are facing for many years ago. The website acts as a communication system, a remote data processing system, an information source, a customer support system, a marketing place, and so on.

2.2.2 *System Branding*

This part has three elements and they all inter-relate. The first is a side-by-side analysis of the problems and the solutions. One of the first principles in communication is if you can describe the problem, you own the solution. So let's say that your problem is the site struggles with duplication and redundant content. The solution is to have authoritative content – which means picking a winner and eliminating redundancy. Once we have identified the problems, we have to think about opposites to identify the solutions. In this case we have to put our list of problems and the corresponding solutions in priority order.

The next branding element is to knit together the side-by-side that compares the company's business goals against the value proposition for the audience and the visual strategy that flows from these.

The third element to branding is our tag line. This is our chance to tie it all together with a memorable phrase. Think of this as the flag we plant on top of the mountain after we have scaled it. This addressed the problem from a user perspective -- that people didn't see the connection between the company's work and their lives – and from a business perspective – that we needed to answer critics who said the company simply didn't work. The visual strategy addressed the goal of being relevant with a friendly look and feel. The business goal was to put the entire organization on the same platform and adopt a new way of doing business. The user's goal was to get a navigation structure that meant no matter what base you were on you wouldn't have to re-learn the navigation to find what you were looking for. The visual strategy was to go for a contemporary look.

2.2.3 Target Audience

Like many organizations we are no stranger to the need to know who the company's site is for. We have to think about the profile of who comes to the site today and who the company want to attract in the future. We can get valuable insights into our audience when we examine the data of the company's Web analytics tool to see where most visitors go, and what they are searching for.

The act of writing down the audience segments and getting buy-in from our Web team can help us channel our energy into being audience-focused. If, on the other hand, we allow this conversation to float and remain unsettled, it hampers our ability to lead and prioritize the workload. Our main concern is to develop a web based application that can satisfy in full the requirements of the following users: Company's mangers, employees, customers, partners, and the public. Each of these users plays an important role in the goals of the company. So taking in count how we can handle their activities is an important issue to consider. We also have to know that what we actually want is not what the audience wants. The content of the website has to be attractive to its users in order to achieve the goals listed in a previous section of this report.

2.2.4 Identify the top tasks

What do people come to the web site to do? After we identify our targets, we have to think through the content and services we have for each group and whether it meets the needs of the users. Typically Web content comes in three buckets:

- Editorial content
- Applications (sometimes split into self-service and job-oriented)
- Instructions

Web sites can do hundreds of different things. But the 80 / 20 rule applies. We are not surprised to find that 80 percent of the audience visits are on 20 percent of the web site pages. When we understand where the activity lies we can concentrate on our the important activities. The rest of it we may find that it is time to retire much of it and re-focus resources on what our audience cares about. In chapter 2 we are going to discuss much on what users can do.

2.2.5 Critical success factors

How will our team's success be judged and rewarded? It is up to us to frame this narrative for those who will make these decisions. If we fail to do this, others may seize this ground from us and tell the wrong story. The risk here is that we will fail in the eyes of our organization and our reputation will suffer.

When we form our roadmap, we need to get it agreed to and signed out by the higher ups. So when we are finished with our design strategy, we have to package it up and present it

to our team and then our boss (MAMWE manager in this case). We have to put an action memo as a cover sheet that lets our boss agree with the document and put their signature on it.

Do we want to be a standards-based organization? Do we want to save money? Do we want to repair or nurture our reputation? We have to think about the impact of these goals both internally and externally. Also we have to make sure that each goal can be measured in some way so we can chart our progress and take a victory lap when we break through our milestones.

The primary goal of a business company is to determine the best ways to increase the profit and occupy the best position in the marketplace. This cannot be accomplished simply by providing the desired services but it needs also to provide them in the appropriate manner in order to satisfy the needs of the customers. Thus how can these objectives be achieved is a key question which involves different factors. First of all the company has to keep an exceptional service in providing electricity and water facilities. And in addition the content of the website is also an important issue to attract people. This is actually we should think about before designing the software.

2.2.6 *The Domain Name System*

When we design a web based system, the name to assign to our application plays an important role. So we should select a unique name which is easy to remember as well as meaningful so as to attract people to use it. Giving a name which is related to the services provided by the company is a good practice. Basically the name you attribute to any web based system is under the control of what is called domain name system or DNS. The **Domain Name System (DNS)** is a hierarchical distributed naming system for computers, services, or any resource connected to the Internet or a private network. It associates various information with domain names assigned to each of the participating entities. A **Domain Name Service** resolves queries for these names into IP addresses for the purpose of locating computer services and devices worldwide. By providing a worldwide, distributed keyword-based redirection service, the Domain Name System is an essential component of the functionality of the Internet.

An often-used analogy to explain the Domain Name System is that it serves as the phone book for the Internet by translating human-friendly computer hostnames into IP addresses. For example, the domain name www.themamwecorporation.km translates to the addresses which is assigned to the web server hosting the **mamwe** application. Unlike a phone book, however, DNS can be quickly updated and these updates distributed, allowing a service's location on the network to change without affecting the end users, who continue to use the same hostname. Users take advantage of this when they recite meaningful Uniform Resource Locators (URLs) and e-mail addresses without having to know how the computer actually locates the services.

2.2.7 Web Scripting

A **scripting language** or **script language** is a programming language that supports the writing of **scripts**, programs written for a software environment that automate the execution of tasks which could alternatively be executed one-by-one by a human operator. Environments that can be automated through scripting include software applications, web pages within a web browser, the shells of operating systems, and several general purpose and domain-specific languages such as those for embedded systems.

Scripts can be written and executed "on-the-fly", without explicit compile and link steps; they are typically created or modified by the person executing them.^[1] A scripting language is usually interpreted from source code or bytecode. By contrast, in the software environment the scripts are written for is typically written in a compiled language and distributed in machine code form; the user may not have access to its source code, let alone be able to modify it.

When writing applications for the web, you can put the programs, or scripts, either on the web server or on the client's browser. While you can put all of the programming on the server, the best approach combines a careful mix of the two. Server-side scripting addresses data management and security, whereas client-side scripting focuses mainly on data checking and page layout. **Server-side scripting** is a technique used in website design which involves embedding scripts in an HTML source code which results in a user's (client's) request to the server website being handled by a script running server-side before the server responds to the client's request.

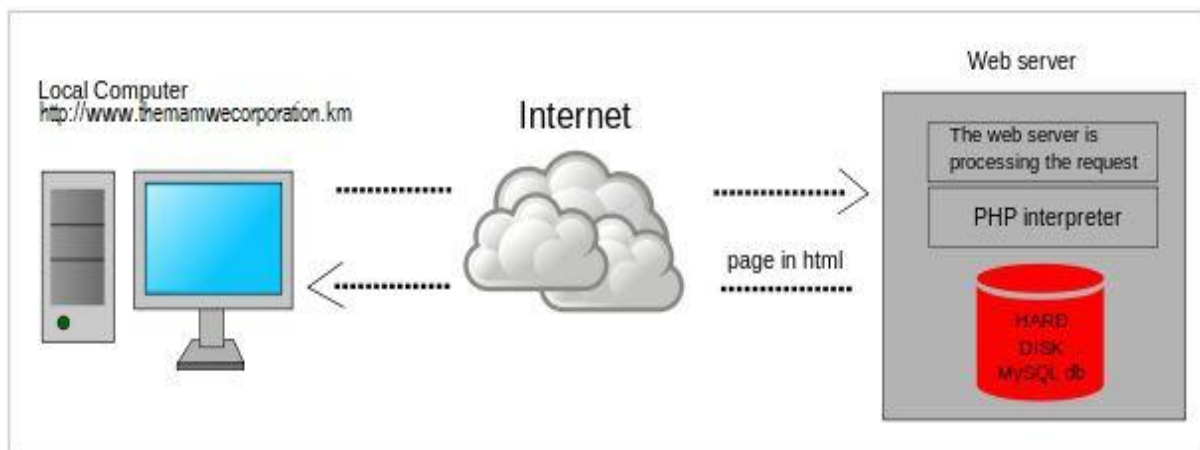


Figure 2.1 : Client Server Scripting Mechanism

The scripts can be written in any of a number of server-side scripting languages available (see below). Server-side scripting differs from client-side scripting where embedded scripts, such as JavaScript, are run client-side in the web browser. Server-side scripting is usually used to provide an interface and to limit client access to proprietary databases or other data sources. These scripts may assemble client characteristics for use in customizing the response based on those characteristics, the user's requirements, access rights, etc. Server-

side scripting also enables the website owner to reduce user access to the source code of server-side scripts which may be proprietary and valuable in itself.

There are a number of server-side scripting languages available, including:

- **ASP** (*.asp, *.aspx)
- **C via CGI** (*.c, *.csp)
- **ColdFusion Markup Language** (*.cfm)
- **Java via JavaServer Pages** (*.jsp)
- **JavaScript using Server-side JavaScript** (*.ssjs, *.js)
- **Lua** (*.lp *.op)
- **Perl CGI** (*.cgi, *.ipl, *.pl)
- **PHP** (*.php)
- **Python via Django** (*.py)
- **Ruby**, e.g. **Ruby on Rails** (*.rb, *.rbw)
- **SMX** (*.smx)
- **Lasso** (*.lasso)
- **WebDNA** (*.dna,*.tpl)
- **Progress WebSpeed** (*.r,*.w)

In the other hand **Client-side scripting** generally refers to the class of computer programs on the web that are executed *client-side*, by the user's web browser, instead of *server-side* (on the web server). This type of computer programming is an important part of the Dynamic HTML (DHTML) concept, enabling web pages to be scripted; that is, to have different and changing content depending on user input, environmental conditions (such as the time of day), or other variables.

Client-side scripts are often embedded within an HTML or XHTML document (hence known as an "embedded script"), but they may also be contained in a separate file, which is referenced by the document (or documents) that use it (hence known as an "external script"). Upon request, the necessary files are sent to the user's computer by the web server (or servers) on which they reside. The user's web browser executes the script, then displays the document, including any visible output from the script. Client-side scripts may also contain instructions for the browser to follow in response to certain user actions, (e.g., clicking a button). Often, these instructions can be followed without further communication with the server. Client-side scripting includes **JavaScript**, **VBScript**, **Perl**, **Tcl/Tk**, and **REXX**.

2.2.8 Web Hosting

The scope of web hosting services varies greatly. The most basic is web page and small-scale file hosting, where files can be uploaded via File Transfer Protocol (FTP) or a Web interface. The files are usually delivered to the Web "as is" or with minimal processing.^[1] Many Internet service providers (ISPs) offer this service free to subscribers. Individuals and organizations may also obtain Web page hosting from alternative service providers. Personal web site hosting is

typically free, advertisement-sponsored, or inexpensive. Business web site hosting often has a higher expense.

2.2.8.1 Website Hosting Steps

2.2.8.1.1 Checking the web hosting information

Here's the important information the Web host should provide:

- **The type of hosting plan.** This includes shared hosting, dedicated hosting, co-location hosting, reseller hosting, template hosting, specialized platform hosting, application hosting, managed services hosting, and high scalability hosting.
- **Cpanel login information** or Web hosting control panel
- **The nameservers**
- **File transfer protocol information**
- **Amount of space and bandwidth**
- **IP address of the web server**
- **Mail server**

2.2.8.1.2 Change your nameservers

Here you need to point the nameservers of your domain name to the nameservers of your web host. If you're using a domain name registrar such as GoDaddy, log into your account, go to the domain name manager to find the domain name you registered with them, then replace the current nameservers with the ones given by your web host.

It will take 24 to 48 hours before your domain name becomes active and enable uploading of your files to the server.

2.2.8.1.3 Upload your files to the server

- Open the FTP software (SmartFTP for example)
- Place your website address in the address box (e.g. themamwecorporation.km)
- Enter the FTP username and password you received from your web host
- Activate the FTP software. This will open 2 document windows
- The first window will show the website files on your computer
- The second window will show the files on your server
- Transfer all your computer files to the server by dragging and dropping them from the first window to the second window

- Make sure you transfer all your files to the public.html folder on the server (this may vary between hosts)

2.2.8.1.4 *Check the appearance of your live website*

Enter your full website address in the browser to view your website. Navigate through all your web pages to check for errors. Make sure there are no broken images, broken links or other design elements missing. Sometimes files don't get properly transferred when they are FTPed to the server.

Important note:

If you're transferring your website from one host to another make sure you back up your website files (including your database and e-mail accounts) before initiating the transfer. This will allow you to return to the original configuration should you make any errors while transferring the files.

2.2.8.2 *Reliability and Uptime*

The availability of a website is measured by the percentage of a year in which the website is publicly accessible and reachable via the internet. This is different than measuring the uptime of a system. Uptime refers to the system itself being online, however it does not take into account being able to reach it as in the event of a network outage.

The formula to determine a system's availability is relatively easy: Total time = 365 days per year * 24 hours per day * 60 minutes per hour = 525,600 minutes per year. To calculate how many minutes of downtime a system may experience per year, take the uptime guarantee and multiply it by total time in a year. In the example of 99.99%: $(1 - .9999) * 525,600 =$ allowable minutes down per year. The following table shows the translation from a given availability percentage to the corresponding amount of time a system would be unavailable per year, month, or week.

Availability %	Downtime per year	Downtime per month*	Downtime per week
90% ("one nine")	36.5 days	72 hours	16.8 hours
95%	18.25 days	36 hours	8.4 hours
97%	10.96 days	21.6 hours	5.04 hours
98%	7.30 days	14.4 hours	3.36 hours
99% ("two nines")	3.65 days	7.20 hours	1.68 hours
99.5%	1.83 days	3.60 hours	50.4 minutes
99.8%	17.52 hours	86.23 minutes	20.16 minutes
99.9% ("three nines")	8.76 hours	43.2 minutes	10.1 minutes
99.95%	4.38 hours	21.56 minutes	5.04 minutes
99.99% ("four nines")	52.56 minutes	4.32 minutes	1.01 minutes
99.999% ("five nines")	5.26 minutes	25.9 seconds	6.05 seconds
99.9999% ("six nines")	31.5 seconds	2.59 seconds	0.605 seconds

Table 2-1: Availability Percentage to Time Conversion

3. System development life cycle

SDLC (System Development Life Cycle), just as the name implies, is defined as the process (as a whole) of developing system or software to meet certain requirements. It covers many activities; starts from understanding why the system should be built, studying the project feasibility, analyzing problems, choosing the system design and architecture, implementing and testing it, up to delivering the system as product to the user. SDLC is a process of *gradual refinement*, meaning that it is done through several development phases. Each phase continues and refines what's done in the previous phase.

3.1 SDLC Phases

Commonly known development phases in SDLC are:

- **Planning.** It is the process of understanding why the system should be built and defining its requirements. It also includes feasibility study from several different perspectives, technical, economic, and organization feasibility aspects.
- **Analysis.** This phase includes activities such as problems identifying and analysis, and even predicting potential problems that may arise in the future regarding the system. The deliverables / products of this phase will drive how the system will be built and guide the developers' works.
- **Design.** System analysis leads to design decision, which exactly determines how the system operates in terms of process, data, hardware, network infrastructures, user interface, and other important factors in the system environment.
- **Implementation.** This is probably the most resource-, cost-, and time-consuming phase of all. This is when the system is actually built, tested, and finally installed. It also includes activities such as user training and system maintenance. Some experts like to separate them into different phases **Deployment** and **Maintenance**. However the above four phases are the most commonly known and accepted steps.

SDLC tries to achieve high quality system that meets or exceeds the requirements. Many methodologies have been developed and introduced in order to implement SDLC. Some of them try to improve other (previously) known methodology. Although each method follows certain different techniques and steps, they all go into the same development phases described above. There are many system development methods known today but most of them are basically extended from three main methodologies which are **Structured Design**, **Rapid Application Development**, and **Object Oriented Analysis and Design**.

3.2 Object-oriented Analysis and Design

The Object-oriented methodology developed based on the lack of synergy between process-centered and data-centered approaches in SDLC. Decomposition of system into a set of process (process centric) or data (data centric) can not be easily obtained, as both aspects are closely related one another. It is difficult to develop system by primarily focusing only to one aspect. As result, the system produced tends to be extendable only in one world. A process centric developed system can not be easily extended when there are changes in type of data in the system. This kind of problems also exists in the data centric developed system.

OO methodology decomposes problems into objects. Objects are considered part of the system that contain both process and data, an object may do some activities/processes (mapped as object methods), an object may also have states (mapped as object attributes). This way, developers will focus on *the entity* in the system that actually does processes and carries data, rather than focus primarily only to one aspect. OO-based system development extensively uses a tool called UML (Unified Modeling Language), which is a set of standard in diagramming and modeling techniques invented by three OO champions, Grady Booch, Ivar Jacobson, and James Rumbaugh, when they worked together in Rational Software. In 1997 UML proposed to and accepted by the Object Management Group (OMG) as a standard diagramming notation in object-oriented system development.

An OO approaches in system development must be:

- Use-case Drive
This means that use-case is the primary modeling tool to define system behavior. Use-cases describe how the users of the system interact with the system to perform activity. And as a use-case focuses only to one activity at a time, it is inherently simple.
- Architecture Centric
The term architecture centric gives a high level view of the system being developed. The software architecture chosen for the system should drive the specification, construction, and documentation of the system itself. The system architecture must support three views of the system:
 - *Functional view*
Describes system behavior from the perspective of users of the system. Use-case diagrams used to depict this functional view.
 - *Static view*
Describes the structure of the system in terms of classes, methods, attributes, and relationships of objects in the system. This view is depicted using CRC (Class Responsibility Collaboration) cards, as well using class and object diagrams.
 - *Dynamic view*
Describes the internal system behavior in terms of object communications and change of states. UML tools used to depict this view are sequence diagrams, collaboration diagrams, and object state-charts.
- Iterative and Incremental

Iterative and Incremental paradigm means that each iteration of the system development must bring the system closer to the requirements. As SDLC is a gradual process, the UML diagrams used in OO-based development moves from a conceptual and abstract thing in the analysis and design phase to become more and more detail in the implementation phase.

4. Tools and Technologies

There are hundreds if not thousands of tools used in web development. In our case the development will be taken though the following tools and technologies:

- *Adobe Dreamweaver CS5*
- *Apache Web Server*
- *Mysql Database*
- *Hypertext Processor or PHP*
- *Java Script*
- *Asynchronous Java Script*
- *JQuery*
- *XmlHTTP*
- *Adobe Photoshop CS3*

Chapter 3 System Analysis and Design

1. Introduction

Before the software coding, it is necessary to analyze the system in order to identify the convenient ways of designing a software system. In the previous chapter, we discussed little bit about the system development life cycle which the most known and standard methodology used in software development. Basically the SDLC discussed above has four main phases. This chapter aims to take us through the analysis of the system being developed. All the four steps may not be explicitly elaborated in the discussion. However the analysis and design which we are going to discuss include all these development phases.

2. Requirements Definition

This stage is the base on which the system development depends throughout the process. It is in this stage where the system is understood. Understanding the system is not easy due to several reasons which we are not going to discuss about in this report. However this understanding can take place in five steps:

2.1 What

Requirements must be determined and agreed to by the customer, the users, and the suppliers. "What" can be defined as follows:

- *What the software must do* to add value for its stakeholders. These functional requirements define the capabilities of the software product.
- *What the software must be* to add value for its stakeholders. These nonfunctional requirements define the characteristics, properties, or qualities that the software product must possess. They define how well the product performs its functions.
- *What limitations there are on the choices* that developers have when implementing the software. The external interface definitions and other constraints define these limitations.

The following figure illustrates the types of requirements and their corresponding levels in the software requirements engineering.

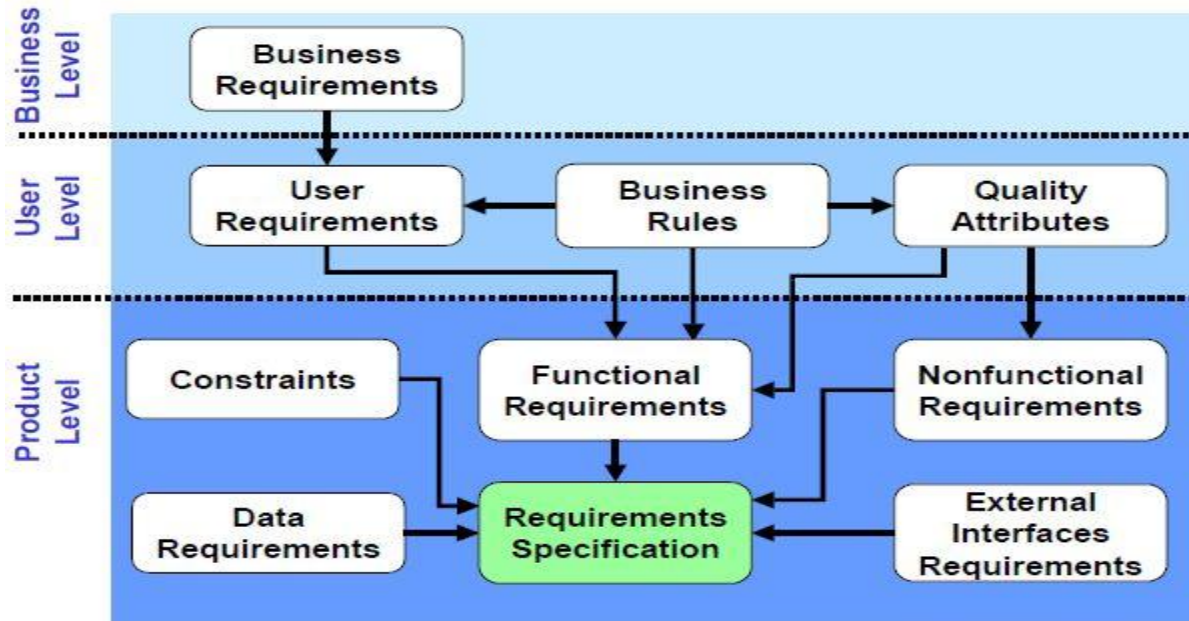


Figure 3.1: Levels of Software Requirements

Business requirements define the business problems to be solved or the business opportunities to be addressed by the software product. In general, the business requirements define why the software product is being developed.

User requirements look at the functionality of the software product from the user's perspective. They define what the software has to do in order for the users to accomplish their objectives. Multiple user level requirements may be needed in order to fulfill a single business requirement as can be shown in the following figure:

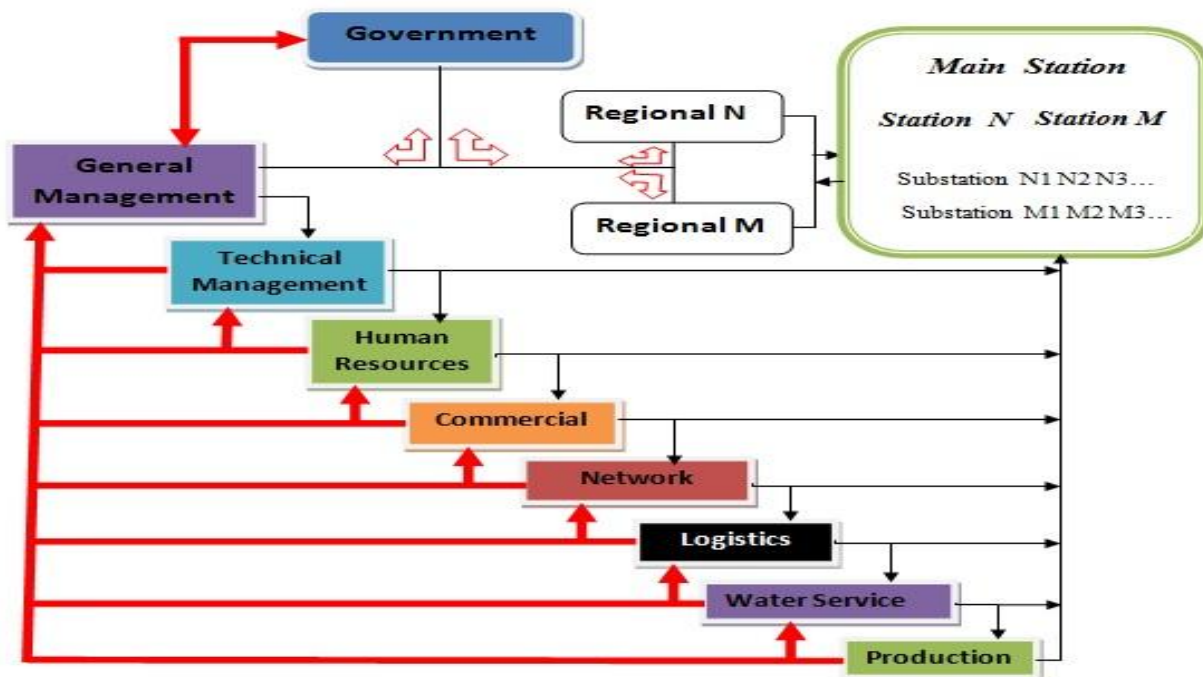


Figure 3.2: User Requirements Versus Organizational Structure

The previous figure shows the different levels of management organized as a department wise structure along with a control management structure. This is not enough for the determination of the user requirement. Therefore the next two figures illustrate the different kinds of users and their corresponding levels in the whole system structure.



Figure 3.3: Organizational Levels of Management

The term “**Levels of Management**” refers to a line of demarcation between various managerial positions in an organization. The number of levels in management increases when the size of the business and work force increases and vice versa. The level of management determines a chain of command, the amount of authority & status enjoyed by any managerial position.

2.1.1 Administrative level

This is referred to **TOP** level in the above figure and it consists of a board of chief executives. The top management is the ultimate source of authority and it manages goals and policies for the company. It devotes more time on planning and coordinating functions.

The role of the top management can be summarized as follows :

- Top management lays down the objectives and broad policies of the company.
- It issues necessary instructions for preparation of department budgets, procedures, schedules etc.
- It prepares strategic plans & policies for the enterprise.
- It appoints the executive for middle level i.e. regional and departmental managers.
- It controls & coordinates the activities of all the departments.
- It is also responsible for maintaining a contact with the outside world.
- It provides guidance and direction.
- It is also responsible towards the shareholders for the performance of the company.

<i>Activity Name</i>	<i>Domain of activity</i>	<i>Recommendation</i>
Installation and Configuration	System wise, rules & flow setup	Strongly
User creation & System init	User categories & System startup	Strongly
Communication	System and outside word	Strongly
Finance	Goods, Services, Projects, Salaries	Strongly
System explorer/Notifications	Medium and Low level activities	Strongly
Statistics (against budget)	Expenses and profit measurements	Recommended
Customer Service	Payment, Transactions, List, etc	Recommended
Reporting	Medium and low level activities	Recommended
Notes taking	Future planning & activities suite	Recommended
Payroll and Scheduling	Employees , Projects , and Payments	Strongly
Engine & Events recorder	Events, Services and Risk assessments	Strongly

Table 3-1: Mamwe Higher Level Management Activities

2.1.2 Executory Level or Middle Level

The regional managers and departmental managers (heads of the departments) constitute middle level. They are responsible to the top management for the functioning of their regional's and departments. They devote more time to organizational and directional functions. In small organization, there is only one layer of middle level of management but in big companies, there may be senior and junior middle level managements. Their role can be emphasized as :

- To execute the plans of the organization in accordance with the policies and directives of the top management.
- To make plans for the sub-units of the organization.
- To participate in employment & training of lower level management.
- To interpret and explain policies from top level management to lower level.
- To are responsible for coordinating the activities within the division or department.
- To prepare important reports and other important data to top level management.
- To evaluate performance of junior managers.
- To be responsible for inspiring lower level managers towards better performance.

Regional Manager		
<i>Activity Name</i>	<i>Domain of Activity</i>	<i>Recommendation</i>
Departmental	Heads, departments, Resources, (or init MIS)	Strongly
Services	Customers, Transactions, Registration, etc	Strongly
Explorer	Complaints, Requests, Notifications	Recommended
Communication	Employees, Other managers, Customers	Strongly
Report / Statistics	Services, Goods, Project, Salaries	Strongly
Scheduling	Payroll, Working Hrs, Service/Project	Strong
Head of Department		
<i>Activity Name</i>	<i>Domain of Activity</i>	<i>Recommendation</i>
Employee	Registration order, Salary and bc information	Strongly
Services	Adding, Assignment, Service Suite, etc	Strongly

Finance	Payment, Transactions, Goods, Services, Project	Strongly
Reporting	Expenses, Complaints, Employees, Services, ...	Strongly
Messaging	Managers, Employees, Outside world	Recommended
Customers	Customer Service, Payments, Complaints,	Recommended

Table 3-2: Mamwe Middle Level Management Activities

2.1.3 Operative Level

Lower level is also known as supervisory / operative level of management. It consists of supervisors, foreman, section officers, superintendent etc. According to *R.C. Davis*, "Supervisory management refers to those executives whose work has to be largely with personal oversight and direction of operative employees". In other words, they are concerned with direction and controlling function of management. Their activities include :

- Assigning of jobs and tasks to various workers.
- They guide and instruct workers for day to day activities.
- They are responsible for the quality as well as quantity of production.
- They are also entrusted with the responsibility of maintaining good relation in the organization.
- They communicate workers problems, suggestions, and recommendatory appeals etc to the higher level and higher level goals and objectives to the workers.
- They help to solve the grievances of the workers.
- They supervise & guide the sub-ordinates.
- They are responsible for providing training to the workers.
- They arrange necessary materials, machines, tools etc for getting the things done.
- They prepare periodical reports about the performance of the workers.
- They ensure discipline in the enterprise.
- They motivate workers.
- They are the image builders of the enterprise because they are in direct contact with the workers.

In fact it is in this level where we can find the largest number of employees. In many cases the users and/or employees in levels 3, 4, and 5 of the next figure are combined together to form a single layer. This is because their activities include a combination of several types of employees. Their role can be summarized as follows:

Registrar		
<i>Activity Name</i>	<i>Domain of Activity</i>	<i>Recommendation</i>
Records keeping	Employees, Customers, Expenses, Events, Products	Strongly
Payroll Summary	For employees attendance	Strongly
Messaging	Heads, Managers, employees, Outside	Recommended
Director		
Team Management	Employees, Others	Strongly
Services	Projects and Services Suite	Strongly

Reporting	Employees and other activities	Strongly
Messaging	Heads, Managers, employees, Outside	Recommended
Scheduling / Budget	Services, Projects	Strongly
Accountant		
Salaries / Payment	Employees	Strongly
Billing Management	Invoice, Payment, Request, Customers	Strongly
Budget, Payment note	Department, Services, Projects, Goods, sales, Payment	Strongly
Reporting	Payments, Expenses, Statistics	Strongly
Messaging	Managers, Heads, Customers, Employees,	Recommended
Engineer		
Planning	Company`s Creativity (for new features)	Recommended
Archive	System activities, Access and Failure rating	Recommended
Reporting/Messaging	Heads, Managers, Directors, Accountant, Salesman	Recommended
Teller		
Window	Customers, Employees	Strongly
Online Service	Customers, Employees	Recommended
Messaging	Customers, Employees	Recommended
Salesperson		
Product Selling	Customers, note Accountant	Strongly
Reporting	Heads, Accountant, Managers	Strongly
Messaging	Heads, Accountant, Managers, Customers	Strongly
Workforce / Public		
Communicating	Teller, Heads, Salesman, Director, Public Users	Strongly
Payment/Purchase	Products, Accountant, Salesman, Heads, Managers	Recommended
Other Events	System, About you, Public	Optional

Table 3-3: Mamwe Lower Level Management Activities

In addition to the above stated levels of management we have identified a more detailed organizational structure as shown in the following figure:

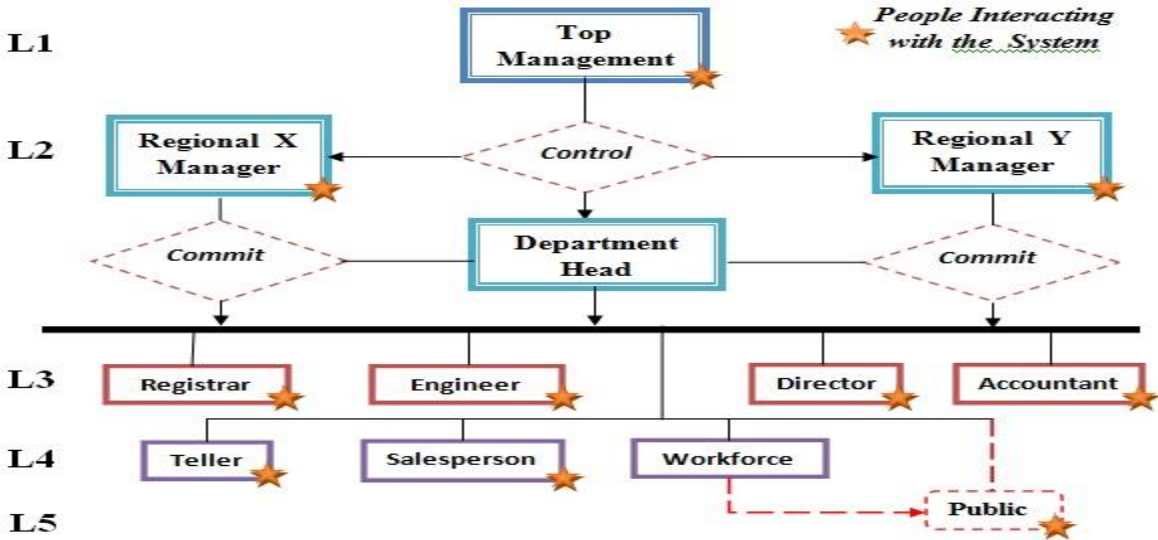


Figure 3.4: Users Organization Structure for Mamwe

An organizational structure as the one shown in the previous figure consists of activities such as task allocation, coordination and supervision , which are directed toward the achievements of the organizational goals. It can also be considered as the viewing glass or perspective through which individuals see their organization and its environment. Organization can be structured in many different ways , depending on their objectives. The structure of an organization will determine the modes in which it operates and performs. Organizational structure allows the expressed allocation of responsibilities for different functions and processes to different entities such as *branch, department, workgroup, and individual*. Organizational structure affects the organizational action in two big ways. *First*, it provides the foundation on which standard operating procedures and routines rest. *Second*, it determines which individuals get to participate in which decision-making processes, and thus to what extent their views shape the organization`s actions. So designing an organization`s structure is a big job. We need to determine the different levels as well as their *relationship and responsibilities*.

In fact the software may be part of a much larger system that includes other components. In this case, the business and user-level requirements feed into the product requirements at the system level. The system architecture then allocates requirements from the set of system requirements downward into the software, hardware, and manual operations components.

2.2 Why

The hardest part of building a software system is deciding precisely what to build. No other part of the conceptual work is as difficult as establishing the detailed technical requirements, including all of the interfaces to people, to machines, and to other software systems. No other part of the work so cripples the resulting system if done wrong. No other part is more difficult to rectify later.

There are many issues that can have a negative impact on software development projects and products if practitioners don't do a good job of defining their software requirements.

These issues include:

- Incomplete requirements
- Lack of user involvement
- Requirements churn
- Wasted resources
- Gold plating
- Inaccurate estimates

If the requirements are incomplete, software practitioners end up building a software product that does not meet all of the customer and user's needs. As illustrated in the following figure, Noritaki Kano developed a model of the relationship between customer satisfaction and quality requirements (Pyzdek 2000). The expected quality line represents those quality requirements that the customer explicitly states.

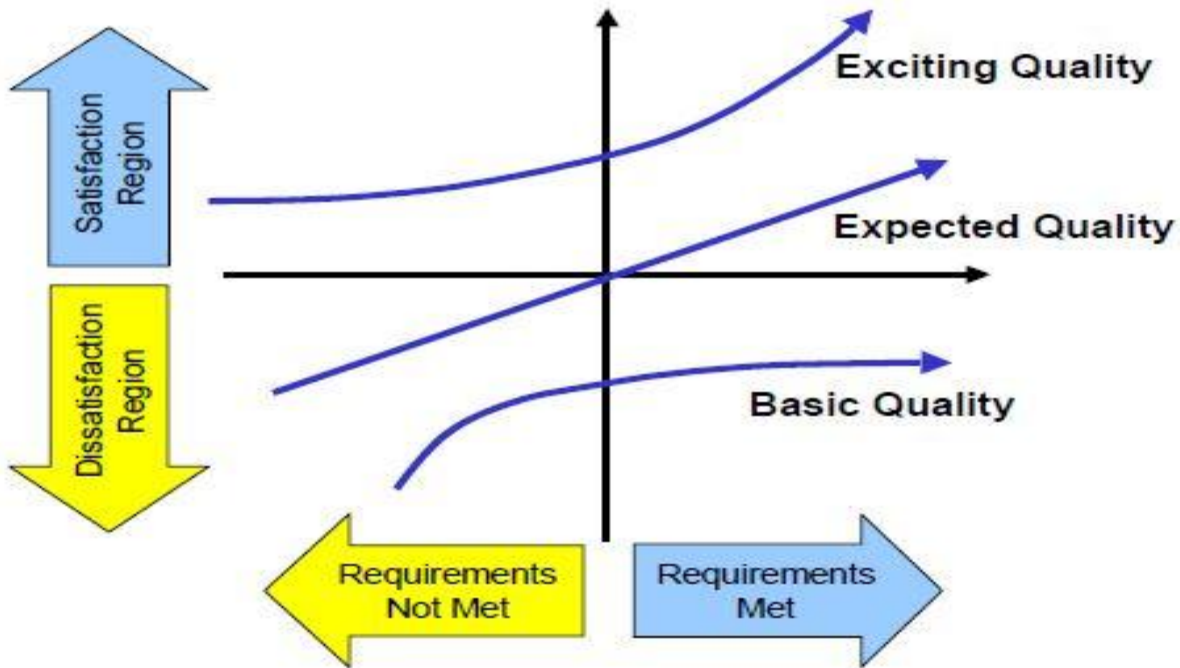


Figure 3.5: Kano Model for Quality Requirements

The expected quality line represents those quality requirements that the customer explicitly states. When enough of their explicit requirements are met, the customer shifts from being dissatisfied with the product to being a satisfied customer. There is a basic level of quality requirements that a customer expects the product to have. These are requirements that are assumed by the customer and are typically not explicitly stated.

The requirements define the scope of the products that are being developed. Without a clear picture of that scope, estimates of the project schedule, cost, and quality will be less accurate.

2.3 Who

Stakeholders are individuals who affect or are affected by the software product and therefore have some level of influence over the requirements for that software product. The requirements engineering process provides the best opportunity to consider all of the various stakeholder's interest in context with one another. There are three main categories of stakeholders: the acquirers of the software product, the suppliers of the software product, and other stakeholders.

The *acquirer* type stakeholders can be divided into two major groups. First there are the customers who request, purchase, and/or pay for the software product in order to meet their business objectives. The second group is the users, also called end-users, who actually use the product directly or use the product indirectly by receiving reports, outputs, or other information generated by the product.

The *suppliers of the software product* include individuals and teams that are part of the organization that develops the software product or are part of the organizations that distribute the software product or are involved in other product delivery methods (for example,

outsourcing). The requirements analyst, also called the business analyst or system analyst, is responsible for eliciting the requirements from the customers, users, and other stakeholders, analyzing the requirements, writing the requirements specification, and communicating the requirements to development and other stakeholders. The designers are responsible for translating the requirements into the software's architectural and detailed designs that specify how the software will be implemented. The developers are responsible for implementing the designs by creating the software products. There may also be *other stakeholders* interested in the requirements.

All these types of stakeholders are represented in the organizational structure shown fig 2.4 of this chapter.

2.4 When

Requirements development encompasses all the activities involved in identifying, capturing, and agreeing upon the requirements. This includes the definition and integration of the business requirements, the user requirements, and software product level requirements. The majority of the requirements development activities occur during the early concept and requirements phases of the life cycle. Continued elaboration of the requirements, however, can progress into the later phase of the software development life cycle.

Requirements management encompasses all of the activities involved in requesting changes to the baselined requirements, performing impact analysis for the requested changes, approving or disapproving those changes, and implementing the approved changes. Requirements management also includes the activities used for ensuring that all work products and project plans are kept consistent and tracking the status of the requirements as one progresses through the software development process. Requirements management is an ongoing activity throughout the software development life cycle.

The implemented software product is validated against its requirements during the testing phases of the life cycle to identify and correct defects and to provide confidence that the product meets those requirements.

Requirements engineering is an iterative process. Practitioners must first develop the business requirements and baseline them. The business requirements are input into the development of the user-level requirements. Based on that effort, practitioners may discover gaps in their business requirements that result in their further refinement. They can then use the information they gain from refining the business requirements for further update of the user-level requirements. The business and user-level requirements feed into the definition of the product-level requirements. This may lead to further refinement of the business and user-level requirements. The product requirements are then input into the software design and development process, which may uncover implicit requirements or the need for further refinement of the business, user, and product-level requirements.

2.5 How

Software requirements engineering is a disciplined, process-oriented approach to the definition, documentation, and maintenance of software requirements throughout the software development life cycle. Requirements development encompasses all of the activities involved in eliciting, analyzing, specifying, and validating the requirements.

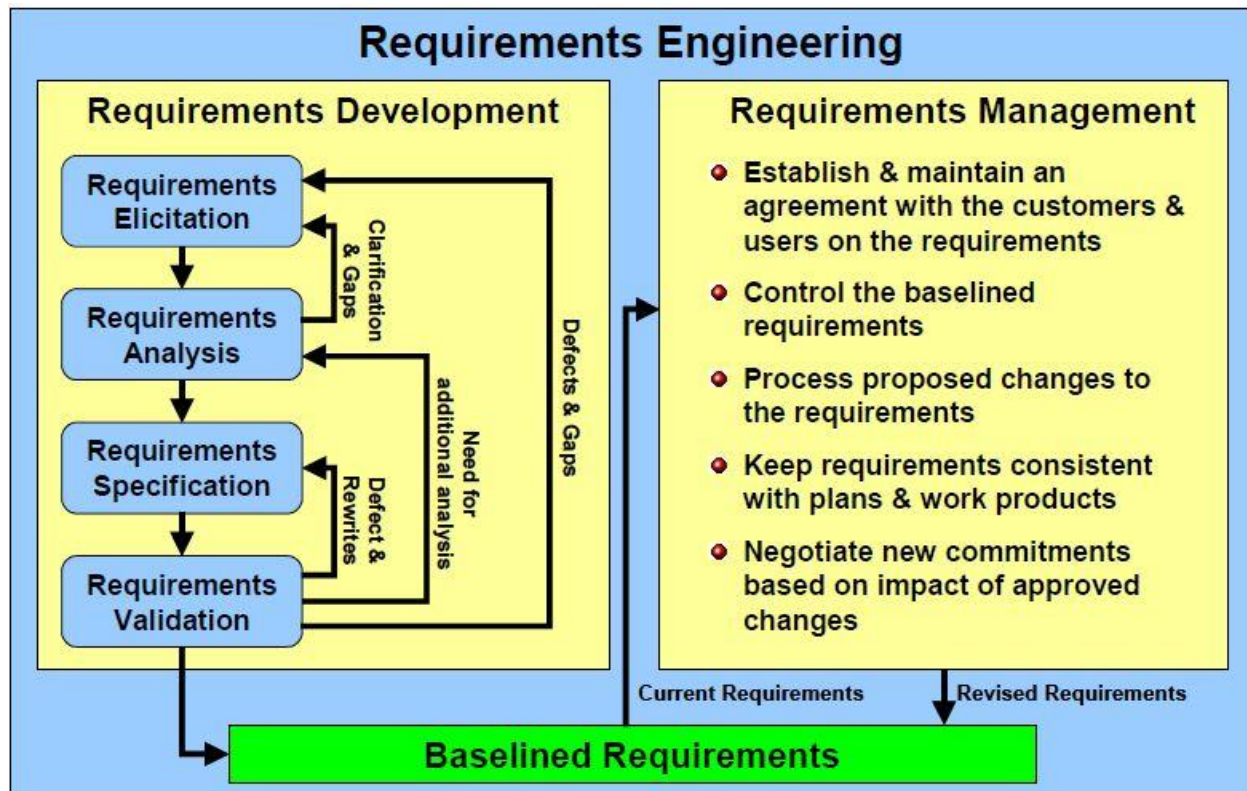


Figure 3.6: Requirements Engineering & Management

The requirements elicitation step includes all of the activities involved in identifying the requirement's stakeholders, selecting representatives from each stakeholder class, and determining the needs of each class of stakeholders. The elicitation process is the information-gathering step in the requirements development process. Many different techniques may be used to elicit requirements, including stakeholder interviews, focus groups, facilitated requirements workshops, observations of current work processes, questionnaires and surveys, analysis of competitor's products, and benchmarking of industry practices. Requirements elicitation may also involve documentation studies of:

- Industry standards, laws, and/or regulations
- Product literature (one's own or the competition's)
- Process documentation and work instructions
- Change requests, problem, or help-desk reports
- Lessons learned from prior projects or products
- Reports and other deliverables from the existing systems

During the requirements analysis step the stakeholder's needs, assumptions, and other information identified during requirements elicitation are melded together and refined into further levels of detail. This step includes representing the requirements in various forms including prototypes and models, performing trade-off analysis, establishing priorities, analyzing feasibility, and looking for gaps that identify missing requirements. The information gained in the analysis step may necessitate iteration with the elicitation step as clarification is needed, conflicts between requirements are explored, or missing requirements are identified. The requirements are formally documented during the specification step so they can be communicated to the product stakeholders. The requirements specification can take one of many forms:

- Business requirements may be documented in a business requirements document (BRD), marketing requirements document (MRD), or project vision and scope document.
- User requirements may be documented in a set of use cases in a tool or in a user requirements specification (URS) document.
- If the software is part of a larger system, the product-level requirements may be documented in a system requirements specification and a system architecture specification may document the allocations of those requirements to the software, hardware, and manual operations components of that system.
- The software functional and nonfunctional requirements and the constraints may be documented in an SRS.
- External interfaces may be included in the SRS or in separate external interface requirements documents.

The last step in the requirements development process is to validate the requirements to ensure that they are well written, complete, and will satisfy the customer needs. Validation may lead one to iterate the other steps in the requirements development process because of identified defects, gaps, additional information or analysis needs, needed clarification, or other issues.

3. Object Oriented Analysis and Design

An object-oriented program may be viewed as a collection of interacting *objects*, as opposed to the conventional model, in which a program is seen as a list of tasks (subroutines) to perform. In OOP, each object is capable of receiving messages, processing data, and sending messages to other objects. Each object can be viewed as an independent "machine" with a distinct role or responsibility. The actions (or "methods") on these objects are closely associated with the object. For example, OOP data structures tend to "carry their own operators around with them"

(or at least "inherit" them from a similar object or class) - except when they have to be serialized.

3.1 UML Diagrams for Interactive System

Unified Modeling Language (UML) is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created, by the Object Management Group. It was first added to the list of OMG adopted technologies in 1997, and has since become the industry standard for modeling software-intensive systems.

UML includes a set of graphic notation techniques to create visual models of object-oriented software-intensive systems. UML is used to specify, visualize, modify, construct and document the artifacts of an object-oriented software-intensive system under development.

During the following discussion we are going to focus on how our system is developed through the oop approach.

Basically not all the processes are included here due to space problem. The first step during this application development is the creation of classes. As you may know software objects belong one or more classes. Therefore the development of classes is then the heart of the system. And not all the classes are discussed here. However our focus is on one class diagram covering all the classes needed for the accounting management of the company.

3.1.1 Accounting module class diagram

The next figure illustrates the interrelationships among all the classes of the diagram. UML 2 considers structure diagrams as a classification; there is no diagram itself called a "Structure Diagram. However, the class diagram offers a prime example of the structure diagram type, and provides us with an initial set of notation elements that all other structure diagrams use.

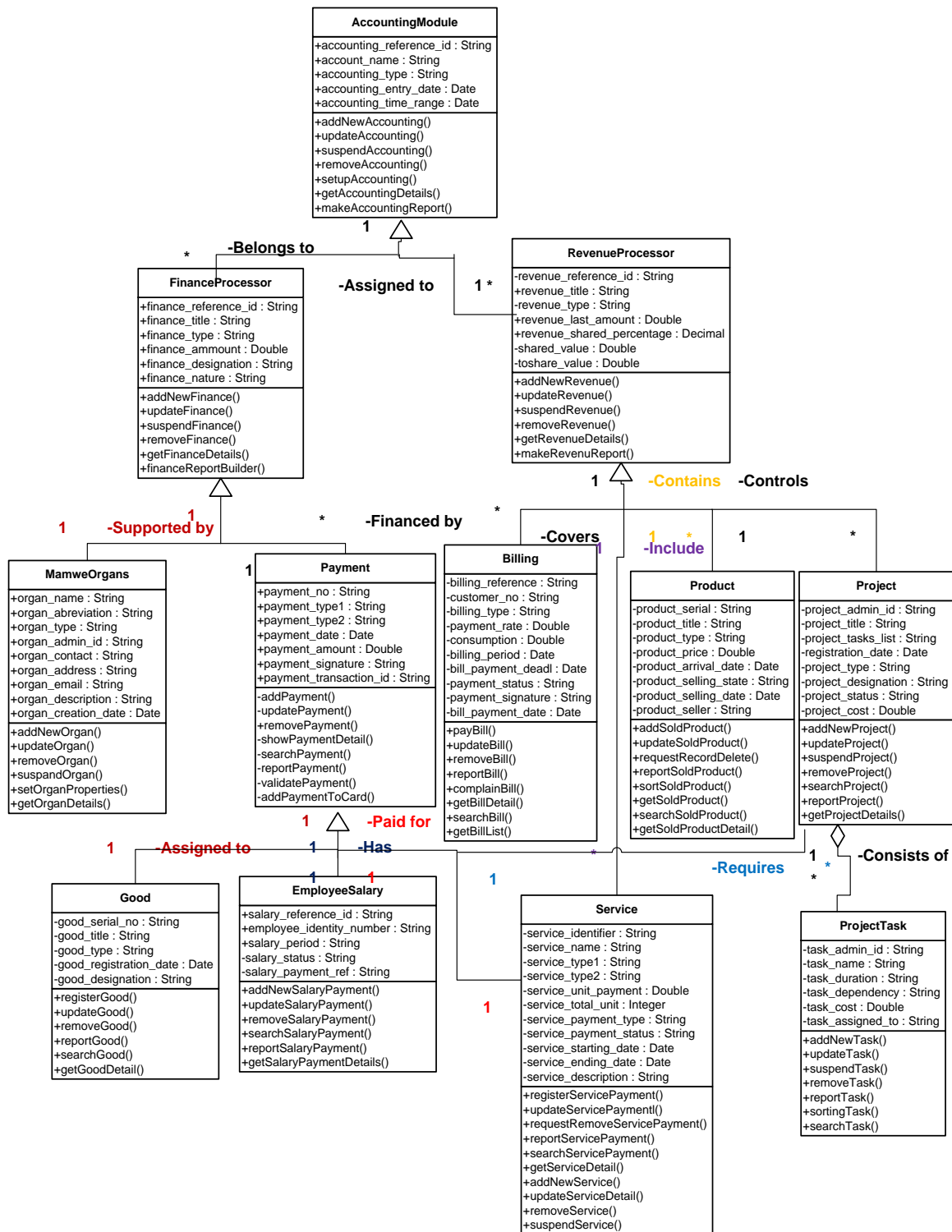


Figure 3.7: Accounting Module Class Diagram

UML 2 class diagrams are the mainstay of object-oriented analysis and design. UML 2 class diagrams show the classes of the system, their interrelationships (including inheritance,

aggregation, and association), and the operations and attributes of the classes. Class diagrams are used for a wide variety of purposes, including both conceptual/domain modeling and detailed design modeling.

3.1.2 Accounting Module Object Diagram

An extension of class diagram is the object diagram. UML object diagrams use a notation similar to class diagrams and are used to illustrate an instance of a class at a particular point in time. You might want to draw an object diagram to illustrate a real-life example of a class and its relationships as shown in the following diagram.

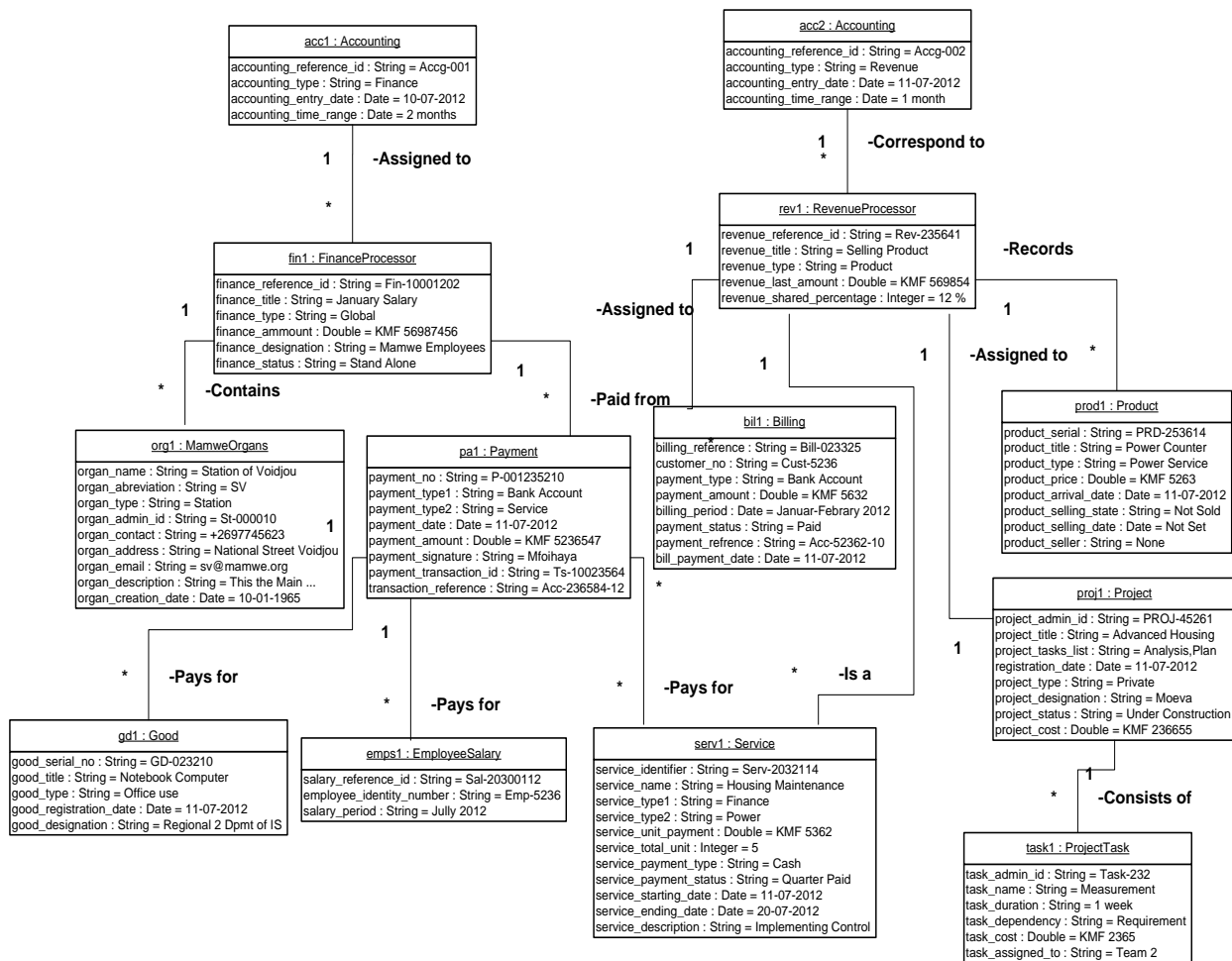


Figure 3.8: Accounting Module Object Diagram

Object diagrams can help clarify classes and inheritance and are sometimes drawn while planning for classes design, or to assist non-programming stakeholders who may find class diagrams too abstract. An object can also be thought of as the description of an individual within a group, while classes describe the characteristics shared by the group. One thing we have to note here is that when we create a new object, called an instance specification, we can assign

an existing class to the object. And more than one instance of an object should be provided wherever we have a many relationship. But in this report we skipped this step and only one instance is created for each and every class.

3.1.3 Some Use Cases of the accounting module

Another UML diagram we used in our development process is the use case diagram. In software and systems engineering, a **use case** is a list of steps, typically defining interactions between a role (known in UML as an "actor") and a system, to achieve a goal. The actor can be a human or an external system. A use case diagram acts as a focus for the description of user requirements. It describes the relationships between requirements, users, and the major components. It does not describe the requirements in detail; these can be described in separate diagrams or in documents that can be linked to each use case.

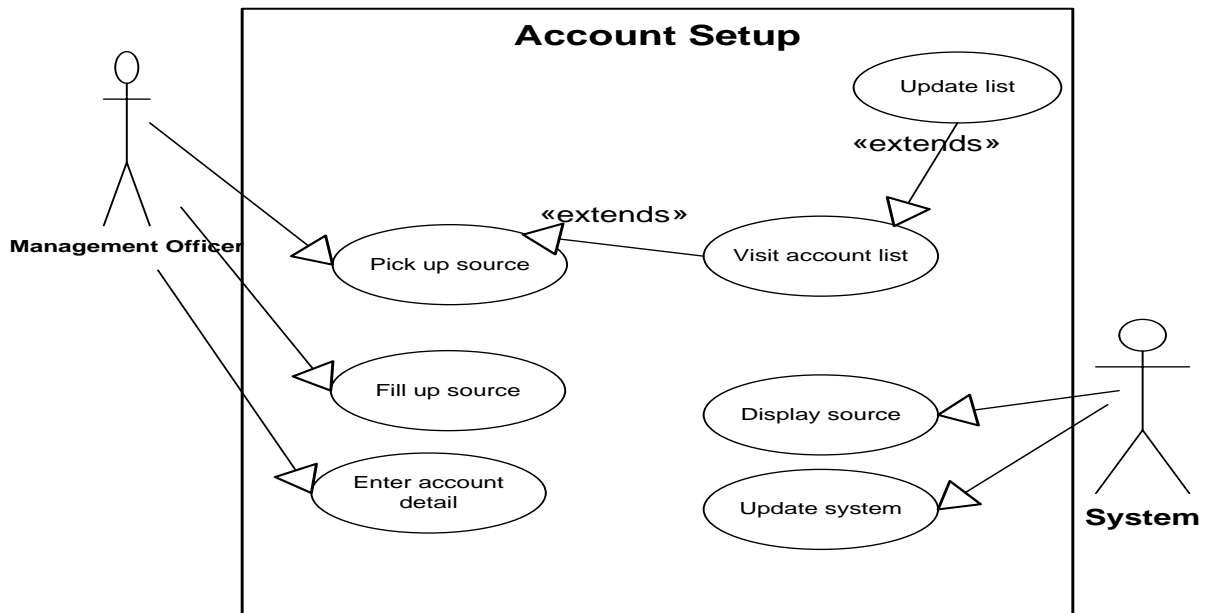


Figure 3.9: Setting Up Account Use Case Diagram

Like in the previous diagram, our discussion is based on the accounting system of the company. But what can an account represent here? Looking only on these figures is not enough to understand the whole picture of the system. Therefore further explanation is needed for this purpose.

In this system, the term account is not same with the bank account that we are familiar with in our daily lives. If you take a look on the class diagram described earlier in this chapter, you can see that the accounting module is divided into two main subsystems namely **FinanceProcessor** and **RevenueProcessor** which are both represented as parent classes of their respective subclasses. Both terms are very complex and need in-depth description using expressions and notations used in the field of finance which are not in the scope of this report. However the

main idea here is very simple. We are using the concept of grouping as illustrated in the following figure.

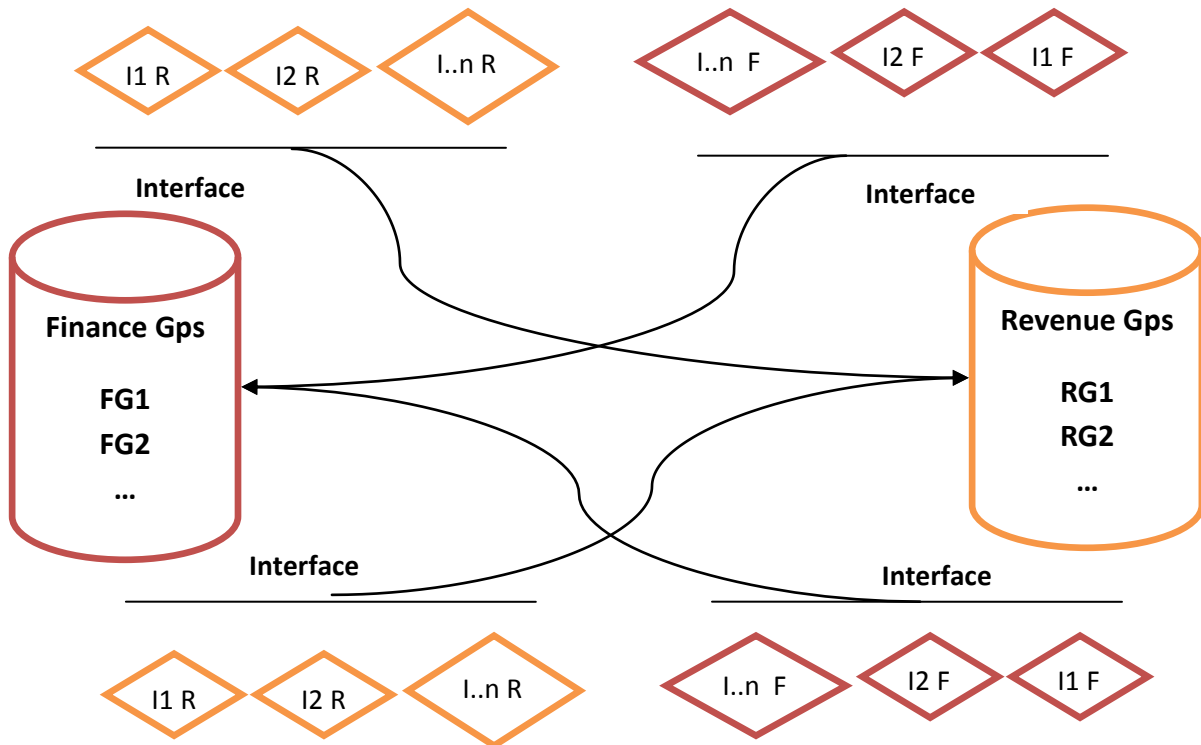


Figure 3.10: Expenses and Revenues to Account Assignment Process

As you can see in this figure, we have two group containers (Finance Groups and Revenue Groups). Each container may have an unlimited number of groups depending on the aims of the manager. Containers receive instances from different interfaces. An interface here is nothing but just a method which can identify between two flags: F for finance and R for revenue. Whenever the interface found the F flag in any object instance, it adds it to the finance groups. Before the instance is assigned to any group inside its parent type (Finance), there is another method or interface which checks whether the incoming instance belongs to any of the existing groups in the same parent. Otherwise an exception will occur. The same process is used for the revenue instances. An example of this process is when the manager wants to add a payment. Payments are of different types such as salary, service, good, product, project, and so on. Therefore these types form the groups in the finance container based on the manager's needs. He may need to organize the expenses for service payment in the month of January separately as illustrated in the next use case diagram. In this case he has to create one account first and assign the appropriate name to that specific account. The same is applicable for all other expense types as well as the revenue types.

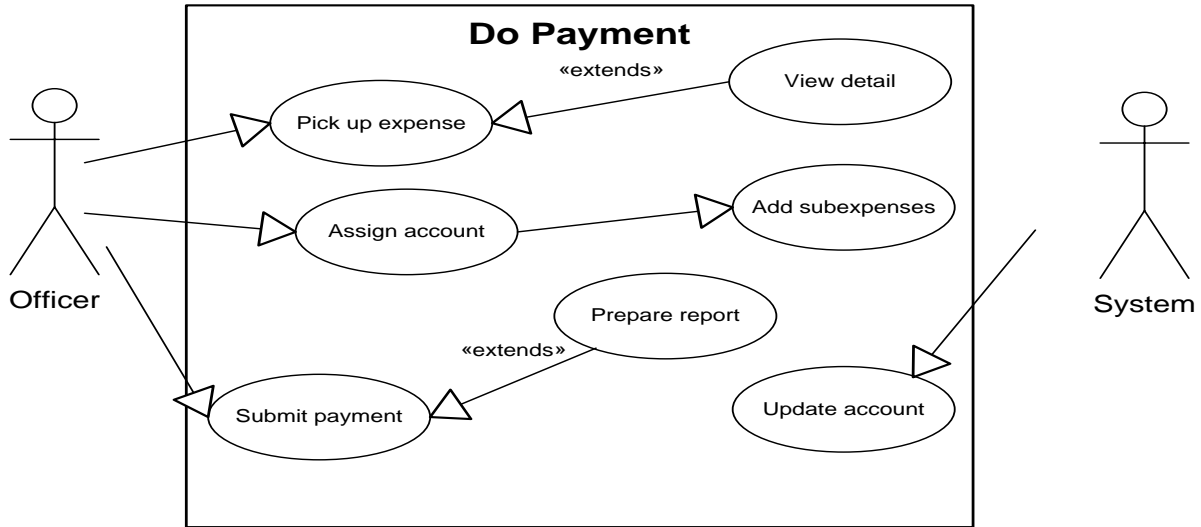


Figure 3.11: Adding Expense/Revenue payment Use Case Diagram

The above use case diagram shows the process of adding a payment in the current payment list. Payment in this case is the as described in the previous use case diagram.

Another important use case we are discussing here is when the manager wants to assign budget for any resource type. The issue is also a main concern of the account types discussed in the previous use cases explanation.

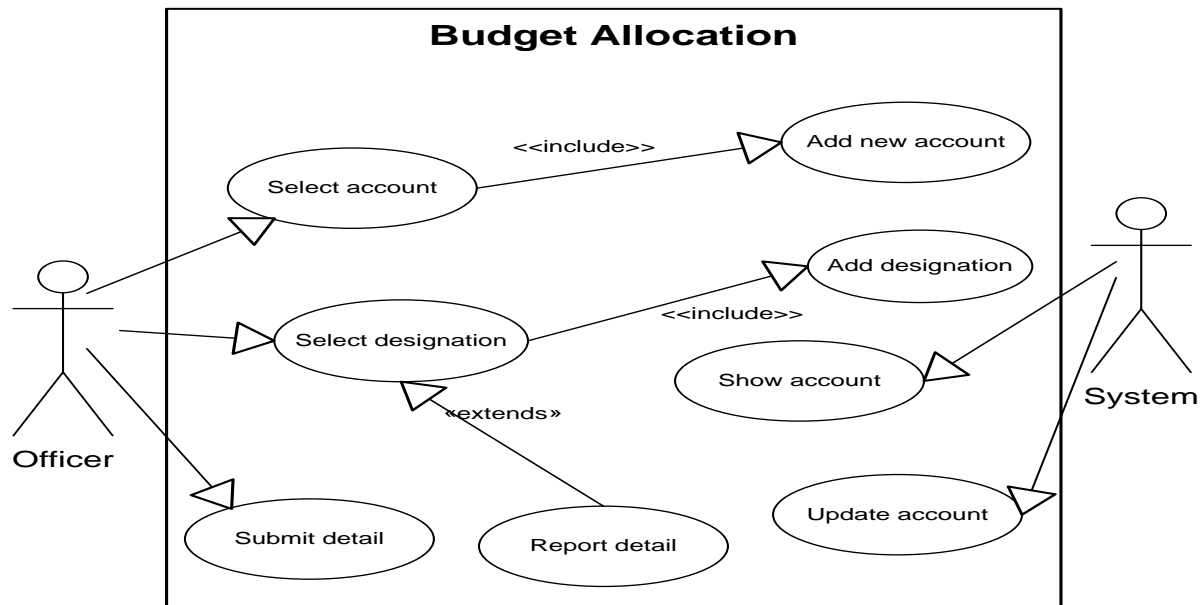


Figure 0.12: Budget Allocation Use Case Diagram

The diagram shows another concept which has not been discussed before. Sometimes there may be exceptions which are undesired events that occur during the process. This can be for

example when the officer has selected an undesirable account because the appropriate account does not exist. In this case he must go back and add an account first. Similar problems may be encountered during the process of performing one task. This will be very much understood in the real life use or testing of the system.

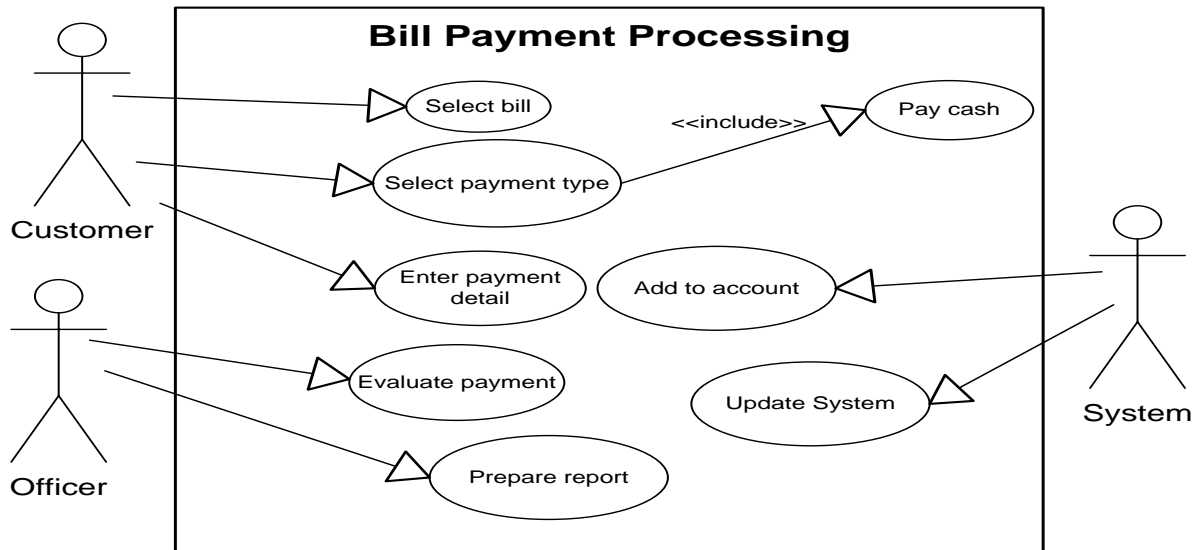


Figure 3.13: Customer Billing Processing Use Case Diagram

The bill payment processing presented in the above figure is completely different from the **Do Payment** task presented in the previous use case. And another type of actor, customer, is included in the process. The process is simple. The customer pays his bill in online. Later on the system administrator or officer will evaluate and finalize the payment process by generating a report or a payment invoice for the customer.

During the implementation of this project, several use cases have been developed for different tasks. However only few use cases are shown in this report. Now for a better understanding of the above use cases we developed a documentation for each use case as shown below.

3.2 Documenting Use case Diagrams

Use Case	<i>Account Setup (creating account content)</i>
Actors	<i>Management Officer, System</i>
Precondition	<i>Account exists or session admin set (to enable creation of accounts)</i>
<p>Flow of events:</p> <ol style="list-style-type: none"> 1. Management officer selects a source type (Finance or Revenue) 2. Before selecting a source type he may visit whether an account exists 3. If no appropriate account exists then update the account list to add a new one 4. Management officer enters the source type details 5. Management officer confirms the account setting by entering the account ID as the main identifier required 6. Manager can repeat the process again and again 7. The system updates the database and should respond to the officer`s request 	
Post condition	<i>Account is filled with account sources (finance and/or revenue)</i>

Table 3-4: Setting Up Account Use Case Documentation

Use Case	<i>Do payment (Expense Payment Processing)</i>
Actors	<i>Management Officer, System</i>
Precondition	<i>Log into Accounting processor , Expense detail available</i>
<p>Flow of events:</p> <ol style="list-style-type: none"> 1. Management officer select an expense source 2. Before proceeding into the next task he may view the expense details first. This is an extension to the payment process 3. If no account was assigned to this expense before then he will do it 4. Manager officer can also allocate funds to the subtasks which have not been assigned any fund from the appropriate account before (project subtasks for example). This is an exception which may occur during the process 5. Management officer submits the payment details 6. Manager officer has the option to prepare any kind of report related to that payment 	

<p>or a number of payments based on the requirements of the manager</p> <p>7. Once the payment is submitted, the system has to update the account from where this payment cost has been subtracted</p>	
Post condition	<i>Payment is ready to be received by its owner</i>

Table 3-5: Adding Expense/Revenue Payment Use Case Documentation

Use Case	<i>Budget Allocation</i>
Actors	<i>Management officer, System</i>
Precondition	<i>Finance account created with predefined amount</i>
<p>Flow of events:</p> <ol style="list-style-type: none"> 1. Management officer selects the account first 2. If the desired account is missing then an exception occurs and he will have to create the appropriate finance account. This happens also if there is a account overflow 3. Under each finance account there may be a number of designations of the fund contained in an existing account. Therefore he will select the designation he wants to allocate the budget. 4. But it may happen that the designation he is looking for does not exist. Then go for creating the designation 5. Before submitting the detail the management officer can print a report. That is an extension in the process 6. Manager officer submits the details once everything is ok 7. The system updates the parent account after the process ends. 	
Post condition	<i>Payment is ready to be received by its owner</i>

Table 3-6: Budget Allocation Use Case Documentation

Use Case	<i>Bill payment processing</i>
Actors	<i>Customer, Management officer, System</i>
Precondition	<i>Customer bill is created and customer logs into his profile</i>
<p>Flow of events:</p> <ol style="list-style-type: none"> 1. Customer will select a bill 	

<ol style="list-style-type: none"> 2. In the payment window the customer clicks on pay and selects a payment type 3. If the customer is not registered for a specific payment type, then he will have to go for a cash payment. In this case the process ends. 4. Otherwise customer enter the payment detail to match with his payment detail stored in the database during his account registration. Then submits the payment. 5. The officer will then evaluate the payment and may prepare report for that or prepare a payment invoice which can be sent over internet. 6. System has to update the concerned account (revenue) and the system as well 	
Post condition	<i>Billing status of the customer is updated</i>

Table 3-7: Bill Payment Processing

4. Conclusion

UML is a general purpose modeling language. It was initially started to capture the behavior of complex software and non software system and now it has become an OMG standard.

UML provides elements and components to support the requirement of complex systems. UML follows the object oriented concepts and methodology. So object oriented systems are generally modeled using the pictorial language. UML diagrams are drawn from different perspectives like design, implementation, deployment etc.

At the conclusion UML can be defined as a modeling language to capture the architectural, behavioral and structural aspects of a system.

Objects are the key to this object oriented world. The basic requirement of object oriented analysis and design is to identify the object efficiently. After that the responsibilities are assigned to the objects. Once this task is complete the design is done using the input from analysis.

The UML has an important role in this OO analysis and design, The UML diagrams are used to model the design. So the UML has an important role to play.

Chapter 4 System Development and Implementation

1. Introduction

Graphical user interfaces (GUIs) have become the user interface of choice. Yet despite the GUI's popularity, surprisingly few programs exhibit good interface design. Moreover, finding information explaining what constitutes a good and intuitive interface is exceedingly difficult. Successful GUIs share many common characteristics. Most importantly, good GUIs are more intuitive than their character-based counterparts. One way to achieve this is to use real-world metaphors whenever possible. Another important characteristic of good GUIs is speed, or more specifically, responsiveness. Many speed issues are handled via the design of the GUI, not the hardware. Depending on the type of application, speed can be the make-or-break factor in determining an application's acceptance in the user community.

2. GUI Design Principles

You can give a GUI the appearance of speed in several ways. Avoid repainting the screen unless it is absolutely necessary. Another method is to have all of the field validations occur on a whole-screen basis, instead of on a field-by-field basis. Also, depending upon the skills of the user, it may be possible to design features into a GUI that give the power user the capability to enter each field of each data record rapidly. The following principles are very important to consider for the development of a good graphical user interface:

2.1 Understand the people

Applications must reflect the perspectives and behaviors of their users. To understand users fully, developers must first understand people because we all share common characteristics. People learn more easily by recognition than by recall. Always attempt to provide a list of data values from which the user can select, rather than having the user key in values from memory.

2.2 Be Careful of Different Perspectives

Many designers unwittingly fall into the perspective trap when it comes to icon design or the overall behavior of the application. I recently saw an icon designed to signify "Rolled Up" totals for an accounting system. To show this function, the designer put a lot of artistic effort into creating an icon resembling a cinnamon roll. Unfortunately, the users of the system had no idea what metaphor the icon was supposed to represent even though it was perfectly intuitive from the designer's perspective. Another point on this issue is the guidelines you should provide to the user. This can be achieved using special message or through a task documentation basis.

Meaning and Behavior	Used to Identify an Application	Used to Identify a Function	Reserved Word Text Label
Information message	No	Yes (identifies an information message box)	None
Warning message	No	Yes (identifies a warning message box)	None
Question message	No	Yes (identifies a question message box)	None
Error message	No	Yes (identifies an error message box)	None

Table 4-1: List of reserved Icons and Messages

2.3 Design for Clarity

GUI applications often are not clear to end users. One effective way to increase the clarity of an application is to develop and use a list of reserved words. A common complaint among users is that certain terms are not clear or consistent. I often see developers engaging in spirited debates over the appropriate term for a button or menu item, only to see this same debate occurring in an adjacent building with a different set of developers. When the application is released, one screen may say "Item," while the next screen says "Product," and a third says "Merchandise", when all three terms denote the same thing. This lack of consistency ultimately leads to confusion and frustration for users. The following figure gives an example of a list of reserved words. An application development group might complete and expand the table with additional reserved words.

Text	Meaning And Behavior	Appears on Button	Appears on Menu	Mnemonic Keystrokes	Shortcut Keystrokes
OK	Accept the data entered or acknowledge the information presented and remove the window	Yes	No	None	Return or Enter
Cancel	Do not accept the data entered and remove the window	Yes	No	None	Esc
Close	Close the current task and continue working with the application; close the view of the data	Yes	Yes	Alt+C	None
Exit	Quit the application	No	Yes	Alt+X	Alt+F4
Help	Invoke the application's help facility	Yes	Yes	Alt+H	F1
Save	Save the data entered and stay in the current window	Yes	Yes	Alt+S	Shift+F12
Save As	Save the data with a new name	No	Yes	Alt+A	F12
Undo	Undo the latest action	No	Yes	Alt+U	Ctrl+Z
Cut	Cut the highlighted information	No	Yes	Alt+T	Ctrl+X
Copy	Copy highlighted information	No	Yes	Alt+C	Ctrl+C
Paste	Paste the copied or cut information at the insertion point	No	Yes	Alt+P	Ctrl+V

Table 4-2: List of Reserved Words

2.4 Design Consistency

Good GUIs use consistent behavior throughout the application and build upon a user's prior knowledge of other successful applications. When writing software for business applications, provide the user with as many consistent behaviors as possible.

2.5 Provide Visual Feedback

If you've ever found yourself mindlessly staring at the hourglass on your terminal while waiting for an operation to finish, you know the frustration of poor visual feedback. Your users will greatly appreciate knowing how much longer a given operation will take before they can enjoy the fruits of their patience. As a general rule, most users like to have a message dialog box with a progress indicator displayed when operations are going to take longer than seven to ten seconds.

There several other principles for GUI design. But basically the number of principles applied depends on the nature of the application as well as the technologies used to develop the software. While many other principles are not in the scope of this report, the above stated principles are enough in our case.

3. Applying Design Principles

Understanding the principles behind good GUI design and applying them to your applications can be a challenge. Let's examine our application to see how these principles can result in an improved interface.

3.1 Home Page interface

Before we go for the home page itself let discuss something about the user access control. First of all any user need to be authenticated in order to make the system more secure. The process starts from the home page where the user will be asked to provide his user name and password. As illustrated in the figure below the authentication process requires a user database record which indicates that he is a registered user.

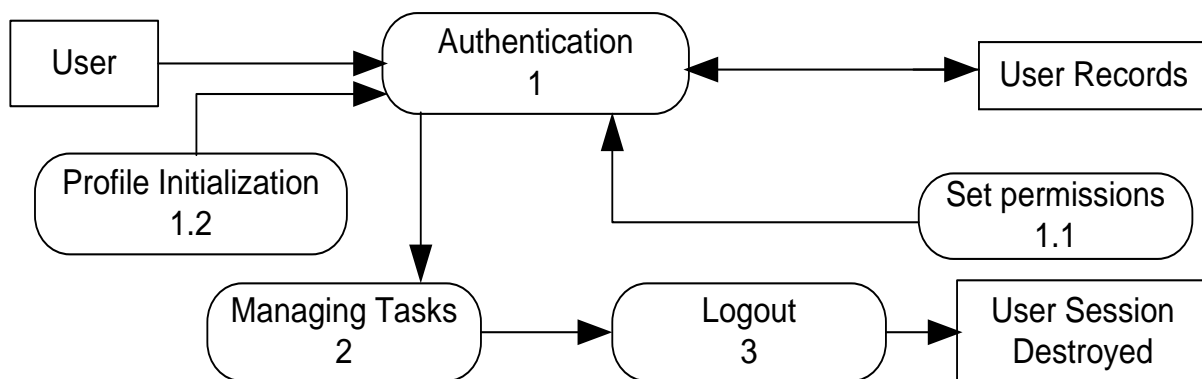


Figure 4.1: User Session Life Time

Once the authentication is confirmed the user is directed into his profile where he will be able to manage any type of task based on the user permissions. Finally when the user logout from the system, the session allocated is destroyed.

3.2 Default Home page



Figure 0.2: Default Application Home Page

A website home page is the gateway through which people get insight into the company services. It does not emphasize on company applications. Rather it just represents what the company is and what is doing. It also allows people to know how the company's customers are supported. Another function of this home page is to provide the public with the abilities to be connected with the outside world by including the media and news links.

3.3 Login Window



Figure 4.3: Home Page With Login Window Tab Activated

3.4 Management Profile Interface

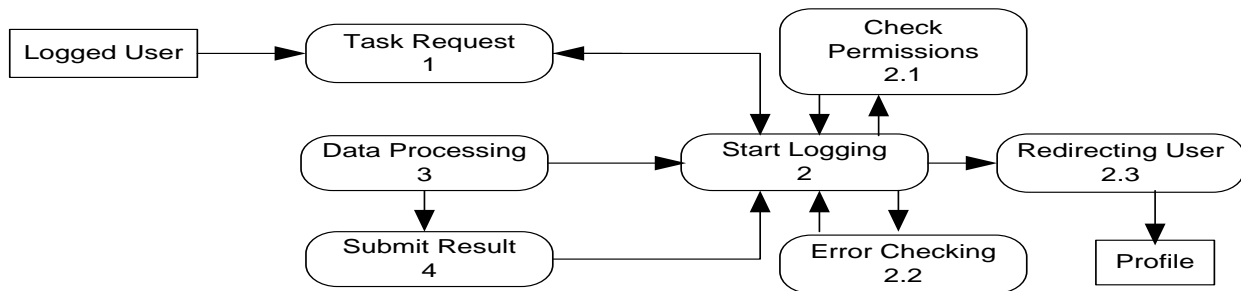


Figure 4.4: Task Processing Life Time

When we were describing the login process presented in figure 17, we did not include any explanation on how a user can perform a task. Yet we have said that the user will be directed to a profile where he may like to perform some tasks. But how a user task can be performed? Well, figure 20 represents the steps through which a user task is performed. First of all we assume that the user is successfully logged in. The after requesting a task(using menu click), many other processes start automatically such as the **logging** whose task is to inform the user about what's happening from the task request to the end of the process, **Check permissions** whose task is to check whether the current user is allowed to perform this task or not, **error checking** whose task is to detect and forward errors to the logging module. Finally when the user submits any result, again the **logging module** starts along with its members processes for making sure that the result submitted is error free or raise a **logging message** otherwise.

3.5 Manager User Guide

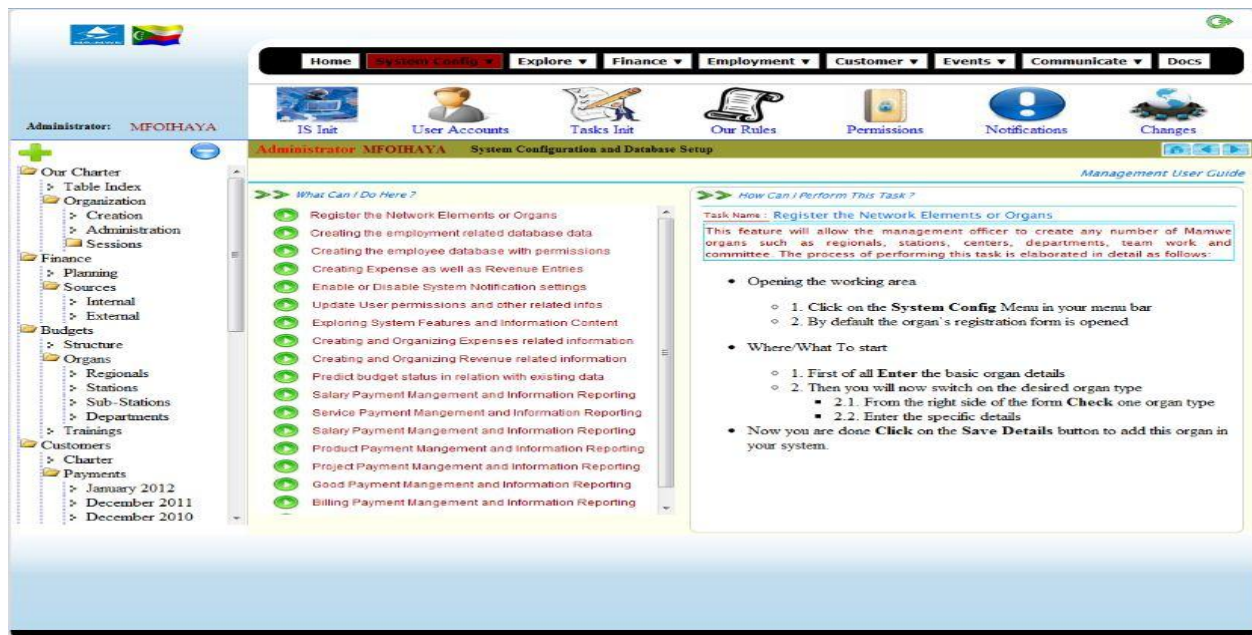


Figure 4.5: Higher Level Management Profile with default user tasks guide

The above discussion is not limited to the management profile only. The same idea is used for all other users. Therefore every user has to be authenticated before he can perform any tasks. Also another important issue of this discussion is that the system can be configured so that all tasks performed can be recoded and saved for the sake of security purpose.

4. Initializing and Configuring the System

Now that we have created the user interfaces, our main concern is to develop a good working environment for performing his duties. However the interfaces only are not enough. We also have to create a database interaction between the user and the system itself. Note that one of the most important objectives of a company's software is to produce reports according to the needs of the company. Other objectives are storing, retrieving, and manipulating data. The main aim of this section is to show and explain different types of interfaces used for the achievements of the above stated objectives.

4.1 Mamwe Organ Registration

A mamwe organ may be of different types: Regional (it is the main head office for one island), Department (a more specific body residing in a regional office which may be given a more specific task), Team Work (a temporary group of employees which is to be given a work to perform during a limited time range).

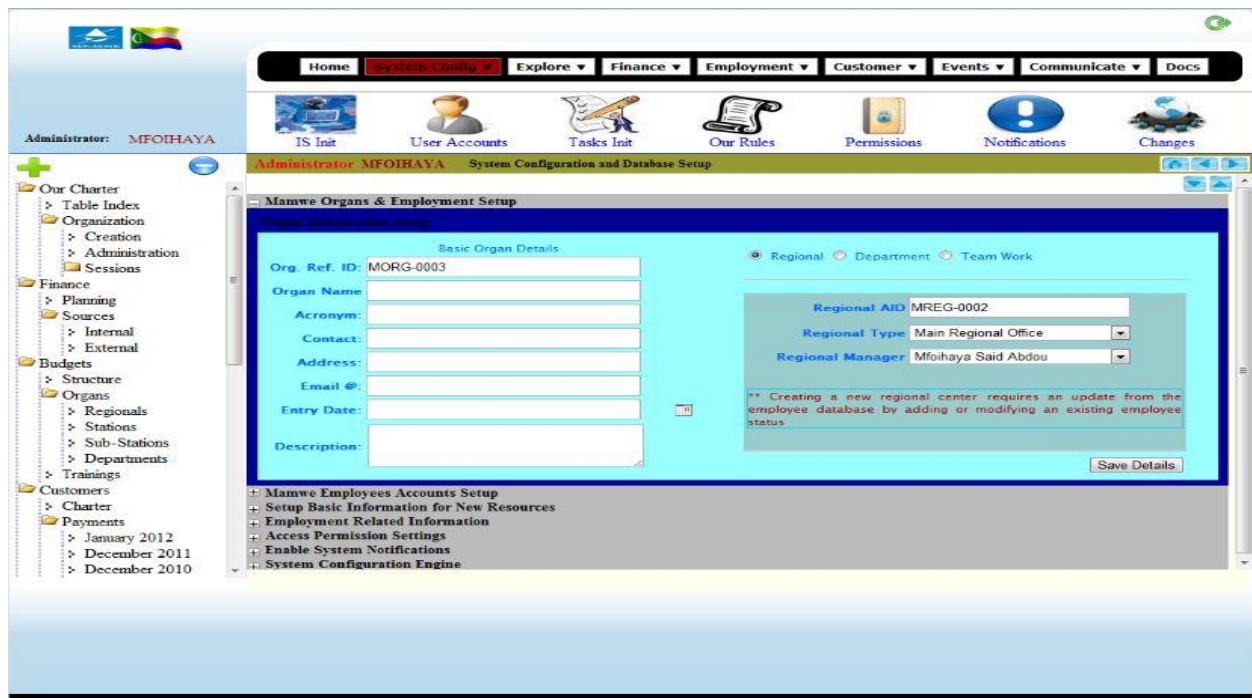


Figure 4.6: Mamwe Organs Registration Form

4.2 Creating Employee Account Record

The screenshot shows a web application interface for creating an employee account record. The interface includes a navigation menu at the top with options like Home, System Config, Explore, Finance, Employment, Customer, Events, Communicate, and Docs. Below the navigation menu are several icons representing different system functions: IS Init, User Accounts, Tasks Init, Our Rules, Permissions, Notifications, and Changes. The main content area is titled "Administrator MFOIHAYA System Configuration and Database Setup" and contains a form for "Mamwe Employees Accounts Setup". The form is divided into several sections: "Mamwe Organs & Employment Setup", "Mamwe Employees Accounts Setup", "User Permissions Settings", and "Setup Basic Information for New Resources". The "Mamwe Employees Accounts Setup" section contains various input fields for employee details, including Individual AID, Full Name, Address, Contact, Date Of Birth, Nationality, Social Security No, Email Address, Gender, Marital Status, Employee No, Regional Office, Department, Function/Task, Join Date, Experience/Yrs, Qualification, and Bank Account. The "User Permissions Settings" section contains several checkboxes for permissions like Read on UqA, Read Write on UqA, Read Write Remove On UqA, Read On HA, Read Write On HA, Read Write Remove On HA, Report Tasks, and Keep Tasks History. The "Setup Basic Information for New Resources" section contains checkboxes for Setup Basic Information for New Resources, Employment Related Information, Access Permission Settings, Enable System Notifications, and System Configuration Engine. The "Account Type" section has radio buttons for Admin, Unique, and Hybrid. A "Save Details" button is located at the bottom right of the form.

Figure 4.7: Employee detail and Accounts Registration

Amount the most important features needed in any company's software is the registration and management of the employees. Here, in figure 23, we have represented the user registration which comprises four categories of details: Personal information, employment detail, account detail, and finally the access permissions. The account detail like account type serves as one identifier of the type of account to be created while access permissions are used to limit the user to a specific task.

4.3 Resources Registration and Management

The screenshot shows a web application interface for resource registration. The top header is green and contains the text 'Administrator MFOIHAYA System Configuration and Database Setup'. Below the header is a sidebar with expandable sections: 'Mamwe Organs & Employment Setup', 'Mamwe Employees Accounts Setup', and 'Setup Basic Information for New Resources'. The main content area has a light blue background and contains a form with the following fields and controls:

- Radio buttons for resource types: Service, Product, Project, Good, Salary, Customer Bill.
- Text input: Service AID (SERV-TPS-0006)
- Text input: Service Name
- Dropdown menu: Service Type 1 (Revenue)
- Dropdown menu: Service Type 2 (Electricity)
- Text input: Unit Payment
- Text input: Total Unit (S)
- Dropdown menu: Payment Mode (Cash Payment)
- Text input: Payment Account
- Text input: Starting Date (with calendar icon)
- Text input: Closing Date (with calendar icon)
- Text input: Description
- Button: Save Details

Below the form is another sidebar with expandable sections: 'Employment Related Information', 'Access Permission Settings', 'Enable System Notifications', and 'System Configuration Engine'.

Figure 4.8: Resources registration & Management Service type Checked

The screenshot shows the same web application interface as Figure 4.8, but with the 'Project' radio button selected. The form fields and controls are:

- Radio buttons for resource types: Service, Product, Project, Good, Salary, Customer Bill.
- Text input: Project Admin ID (PROJ-TPS-0004)
- Text input: Project Name
- Text input: Project Tasks List
- Text input: Registration Date (with calendar icon)
- Text input: Closing Date (with calendar icon)
- Dropdown menu: Project Type (In Company)
- Text input: Designation
- Text input: Project Status
- Button: Save Details

The sidebar sections remain the same as in Figure 4.8.

Figure 4.9: Resources Registration & management Project type checked

4.4 Generating Salaries

Figure 4.10: Resources registration & management Salary type checked with default generator

Employee Name	Function Title	Basic Salary
Mfoihaya Said Abdou	General Manager	KMF 636500
Mansoib Toibibou	Department Head	KMF 869356
Hasan Soilahoudine	Assistant General Manger	KMF 574000
Ahmed Abdul Majid	Department Head	KMF 869356
Fatima Abdou Mfoihaya	General Manager	KMF 636500

Total found 5 **Total Cost** KMF 3585712

What am i doing ?

The request submitted now shows that you want to generate salary for all the Mamwe employees. If you **continue** the cost shown in the left side will be applied. However all the records that will be generated here will be in a waiting state till their corresponding payments are signed by an administrator. Click on **Generate Now** button to continue.

Figure 4.11: Resources registration & management Salary type checked - Generating all salaries

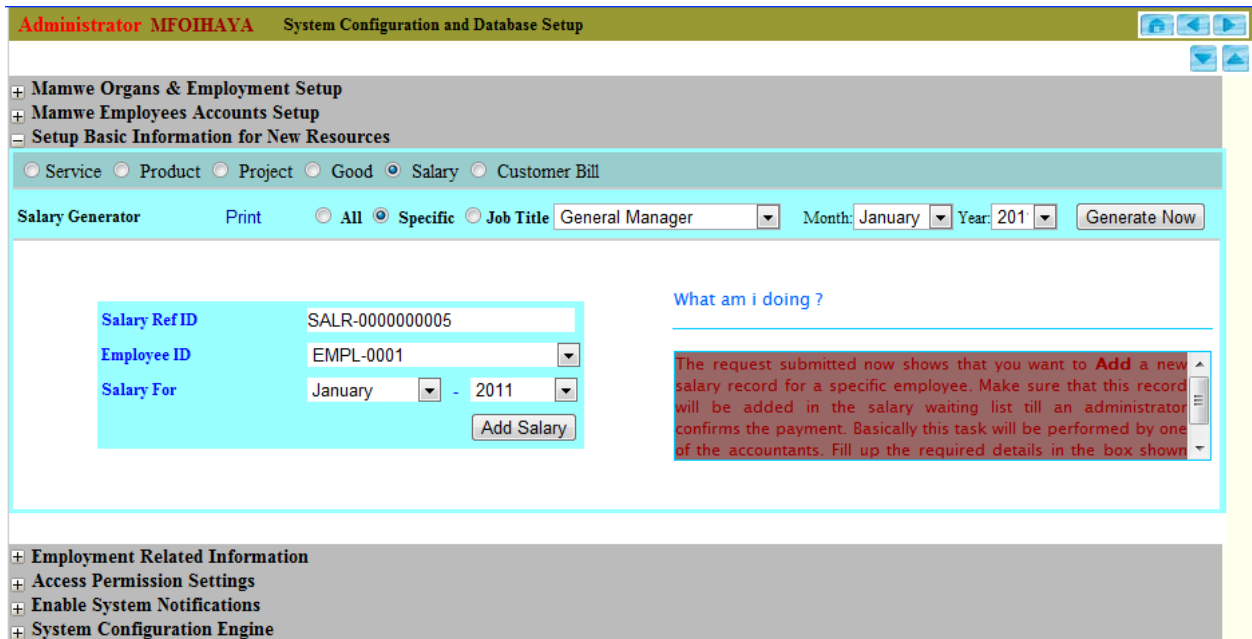


Figure 4.12: Resources registration & management Salary type checked - Generating a specific salary

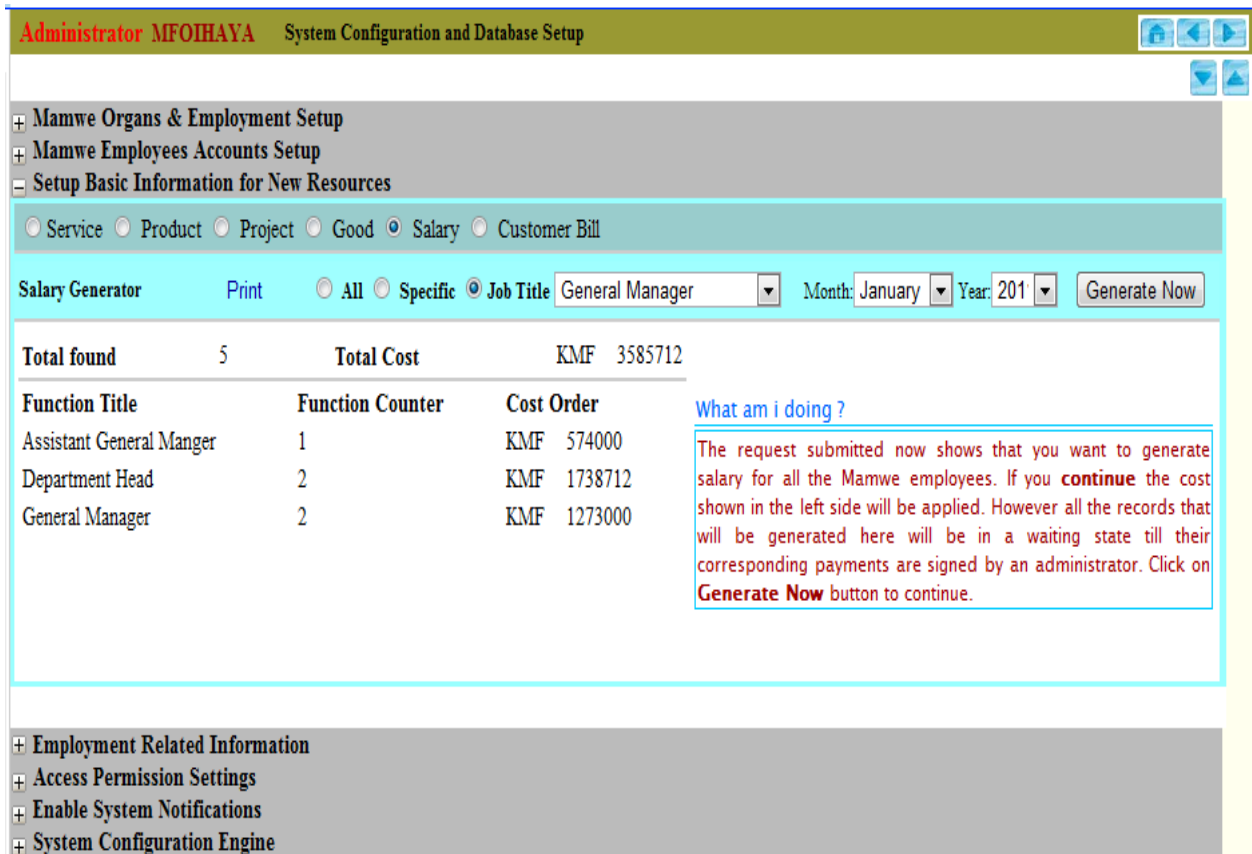


Figure 4.13: Resources registration & management Salary type checked - Generating Job based Salaries

4.5 Creating Customer Billing Record

Administrator MFOIHAYA System Configuration and Database Setup

- + Mamwe Organs & Employment Setup
- + Mamwe Employees Accounts Setup
- Setup Basic Information for New Resources
 - Service
 - Product
 - Project
 - Good
 - Salary
 - Customer Bill
- + Employment Related Information
- + Access Permission Settings
- + Enable System Notifications
- + System Configuration Engine

Add New MAMWE Bank Account Account Name : Account No : - -

Billing Serial No Pay To Account

Customer Number Payment Rate

Customer Bill Type Consumption

Payment Deadline Billing Period To 201

Figure 4.14: Resources registration & management Customer Billing Creator

4.6 Setting up Employment Information System

Administrator MFOIHAYA System Configuration and Database Setup

- + Mamwe Organs & Employment Setup
- + Mamwe Employees Accounts Setup
- + Setup Basic Information for New Resources
- Employment Related Information
- + Access Permission Settings
- + Enable System Notifications
- + System Configuration Engine

New Job Title & Details

Occupation Title

Occupation Salary

Occupation Career

Occupation Center

Payment Mode

New Payment Mode Detail

P. Mode Admin ID

Payment Mode Title

Payment Mode Desc.

Set as Default Mode

Figure 4.15: Setting up employment related information

4.7 Creating a Finance Account Type

Administrator MFOIHAYA Accounting and Finance processing

ACCOUNTS PAYMENT EXPENSES TRANSACTIONS CUSTOMER BILLING SALES STATISTICS REPORT

New Account Source Finance Revenue

Accounting Grouping Manager

Finance AID: FINC-0005
 Title: _____
 Type & Amount: Global - KMF
 Designation: _____
 Nature: Stand Alone

Creating New Account

Reference No: MACC-0003
 Account Name: _____
 Account Type: Financial
 Creation Date: _____
 Expiry Date: _____

Account Name	Account Type	Closing Date
Semester 1 2012 projects and Services	Financial	10-7-2013
Jan-Feb Customer Billing 2012	Revenue	29-2-2012
Revenue Title	Revenue Type	Actual Amount
Customer Payment for the month of October 2011	Customer Billing	269020.5818
First Semester 2012 Private Services	Service	0
Customer Sold products 2012	Product	0
Water Service Private Projects 2012	Project	0
Finance Title	Finance Type	Actual Amount
Construction of power network in north Moheli	Project	0
January session finance 2013	Global	0
January-August Salaries 2012	Salary	328000
Office Use Facilities 2012	Good	0

Figure 4.16: Finance Accounts Creation Process

4.8 Creating a Revenue Account Type

Administrator MFOIHAYA Accounting and Finance processing

ACCOUNTS PAYMENT EXPENSES TRANSACTIONS CUSTOMER BILLING SALES STATISTICS REPORT

New Account Source Finance Revenue

Accounting Grouping Manager

Revenue Ref No: REVN-0005
 Revenue Title: _____
 Revenue Type: Customer Billing
 Sharing Level: % 00

Creating New Account

Reference No: MACC-0003
 Account Name: _____
 Account Type: Financial
 Creation Date: _____
 Expiry Date: _____

Account Name	Account Type	Closing Date
Semester 1 2012 projects and Services	Financial	10-7-2013
Jan-Feb Customer Billing 2012	Revenue	29-2-2012
Revenue Title	Revenue Type	Actual Amount
Customer Payment for the month of October 2011	Customer Billing	269020.5818
First Semester 2012 Private Services	Service	0
Customer Sold products 2012	Product	0
Water Service Private Projects 2012	Project	0
Finance Title	Finance Type	Actual Amount
Construction of power network in north Moheli	Project	0
January session finance 2013	Global	0
January-August Salaries 2012	Salary	328000
Office Use Facilities 2012	Good	0

Figure 4.17: Revenue Accounts Creation Process

4.9 Assigning Expenses and Revenues to Accounts

Figure 4.18: Finance & Revenue to Accounts Assignment Process

This figure represents the most complex and useful tool of this software. When the manager wants to organize the in and out flows of money, he will take advantage of this tool. This function is an application of the concept represented in figure 13 where we represented the out flow with the flag F(finance) and the in flow with the flag R(Revenue). There can be several finance accounts as for the revenue. The main idea is to group either flow based on a period of time. For a revenue account for example, the manager may need to know how much we earn in this period as compared for another period which also has an independent account. Revenue is in different forms such as customer billing, selling products, customer assistance, private service, and so on. Similarly a finance account holds different expenses in a timely based manner. The difference between these two account types is based on the initial value of the account. For a finance account the maximum amount to be allocated has to be assigned during the creation of the account whereas a revenue account is created with amount zero and will be updated once a revenue has been added into the account.

Implementing such a function provides a great number of advantages including the following:

- Measuring Expenses and Revenues
- Help in decision making
- Evaluation of the company's budget
- Prevention and planning

- Transparency and simplicity in fund processing

5. Building Reports

5.1 Building a Salary Payment Report

Administrator MFOIHAYA Accounting and Finance processing

ACCOUNTS PAYMENT EXPENSES TRANSACTIONS CUSTOMER BILLING SALES STATISTICS REPORT

Salaries

SALR-0000000002	March2012
SALR-0000000003	May2012
SALR-0000000004	January2012

Services

Products

Projects

Goods

Customer Billing

Pay Now Print

MADJI-NA-MWENJE ZA KOMORI
Salary Payment Detail Report

Payment Request for SALARY: SALR-0000000002 Date: Friday 07th September 2012

Employee No: EMPL-0004 Employee Name: Ahmed Abdul Majid

Designation: Department Head Career: Career Administrative

Regional Office: Direction Regionale de Ngazidja / Department of Information System

Payment Mode	Account No	Due Salary
Bank Account	36254-584555665211-695584	869356

Payment No: PMNT-0000000001 Finance Ref No: Finance Ref does not exist

P.Type 1: Bank Account Amount: 869356

P.Type 2: Salary Signed By: mfoihaya

P.Source: SALR-0000000002 Transaction: TRAN-0000000001

Preview Pay Now

Figure 4.19: Adding New Salary Payment & Report Making

5.2 Building a Service Report

Administrator MFOIHAYA Accounting and Finance processing

ACCOUNTS **PAYMENT EXPENSES** TRANSACTIONS CUSTOMER BILLING SALES STATISTICS REPORT

Salaries
Services
SERV-Rev-Ele-0003
SERV-Rev-Ele-0002
SERV-Fin-Ele-0003
SERV-Fin-0004
SERV-Fin-0005
Products
Projects
Goods
Customer Billing

Pay Now Print

MADJI-NA-MWENJE ZA KOMORI
Service Detail Report

Payment Request for SERVICE : SERV-Rev-Ele-0002 Date Friday 07th September 2012

Service Name	Service Type 1	Service Type 2
House Wiring Planing and Design	Revenue	Electricity
Registered On 5-8-2011	To start on 12-10-2011	To be closed on 25-1-2012

Service Designation: This service is requested by the SunTek house builder Company

Due Payment (KMF): 10520 Pay To : 5632-859632541-96358

Payment Ref No PMNT-0000000001 Revenue Ref No Revenue Ref does not exist
P.Type 1 Any Amount 10520
P.Type 2 Service Signed By mfoihaya
P.Source SERV-Rev-Ele-0002 Transaction TRAN-0000000001

Preview Pay Now

Figure 4.20: Registering service payment & report making

Both previous figures(35 and 36) illustrates a similar task. However figure 35 shows the salary payment and report making while figure 36 illustrates the service payment and report building.

5.3 Building Bill Payment report

MADJI-NA-MWENJE ZA KOMORI
Customer Bill Payment Report

Billing Identity
 Payment Request for CUSTOMER BILL : BILL-000000002
 Pay To Account : 5236-755855142555-85697
 Date: Friday 07th September 2012

Customer Detail
 Customer No: CUST-0002
 Customer Name: Loulou mohamed

Bill Payment Detail		
Billing type	Billing Period	Consumption
Electricity	June-August-2012	52364
Payment Deadline	Payment Rate	Due Payment
31-8-2012	20.3365	1064900.486

Payment No: PMNT-000000001
 Revenue Ref No: Revenue Ref does not exist

P.Type 1: Bank Account
 Amount: 1064900.486

P.Type 2: Billing
 Signed By: mfoihaya

Source ID: BILL-000000002
 Transaction: TRAN-000000001

[Pay Now](#)

Figure 4.21: Customer Billing Confirmation & Report Making

5.4 Report Printing Options

Print
 Total: 1 page
 Save Cancel

Destination Save as PDF
 Change...

Pages All
 e.g. 1-5, 8, 11-13

Layout Portrait
 Landscape

Margins

Options Headers and footers
 Print using system dialog... (Ctrl+Shift+P)

MADJI-NA-MWENJE ZA KOMORI
Customer Invoice Print Preview

Billing Identity
 Payment Request for CUSTOMER BILL : BILL-000000002
 Pay To Account : 5236-755855142555-85697
 Date: Friday 07th September 2012

Customer Detail
 Customer No: CUST-0002
 Customer Name: Loulou mohamed

Bill Payment Detail		
Billing type	Billing Period	Consumption
Electricity	June-August-2012	52364
Payment Deadline	Payment Rate	Due Payment
31-8-2012	20.3365	1064900.486

Payment No: PMNT-000000001
 Revenue Ref No: Revenue Ref does not exist

P.Type 1: Bank Account
 Amount: 1064900.486

P.Type 2: Billing
 Signed By: mfoihaya

Source ID: BILL-000000002
 Transaction: TRAN-000000001


[Pay Now](#)

Figure 4.22: Customer Invoice Print Preview


Figure 37 shows how making a customer billing payment is done. The process requires the existence of a billing information, customer detail, and finally a payment detail. In figure 38 we are showing the preview of a report or billing receipt print.

5.4 Printed Report Example

9/11/12



MADJI-NA-MWENJE ZA KOMORI
Manwe Actual Economy Statistics



Date: Tuesday 11th September 2012

Actual Revenue

Revenue Source	Source Identifier	Paid Amount	Payment Date
Billing	BILL-0000000002	KMF 1064900.486	11-09-12
Total KMF			1064900.486

Actual Finance

Finance Source	Source Identifier	Paid Amount	Payment Date
Salary	SALR-0000000001	KMF 869356	10-09-12
Salary	SALR-0000000001	KMF 869356	10-09-12
Salary	SALR-0000000008	KMF 574000	10-09-12
Salary	SALR-0000000007	KMF 869356	10-09-12
Salary	SALR-0000000006	KMF 636500	10-09-12
Salary	SALR-0000000009	KMF 869356	10-09-12
Salary	SALR-0000000010	KMF 636500	10-09-12
Total KMF			5324424

Waiting Revenue

Source	Source Identifier	Waiting Amount	Entry Date
Billing	BILL-0000000001	KMF 269020.5818	12-6-2012
Billing	BILL-0000000003	KMF 927995.168	20-11-2012
Total KMF			1197015.7498

Waiting Expenses

Source	Source Identifier	Waiting Amount	Entry Date
Salary	SALR-0000000002	KMF 869356	March2012
Salary	SALR-0000000003	KMF 1505856	May2012
Salary	SALR-0000000004	KMF 2079856	January2012
Salary	SALR-0000000005	KMF 2949212	September2012
Total KMF			2949212

1/1

Figure 4.23: Customer Billing Printed Invoice Sample

Producing and printing customer billing receipts is one of the primary daily needs of the company. Usually when a customer paid a bill, he has to be provided with a paper containing all the necessary information including customer details, billing details, and payment details.

Chapter 5 Customer Support and Feedback

1. Overview

The Release and Maintenance phase is the "support phase" for a system. Defects may be discovered that could push the system back into the Construction and Assembly phase to implement fixes. New requirements may also be requested and documented that could lead to future iterations, taking the system back to the Requirements and Analysis phase.

2. Post Implementation Review

The key to a successful PIR is recognizing that the time spent on the project is just a small part of an ongoing time-line. For people and organizations that will be working on similar projects in the future, it makes sense to learn as many lessons as possible, so that mistakes are not repeated in future projects. And for organizations benefiting from the project, it makes sense to ensure that all desired benefits have been realized, and to understand what additional benefits can be achieved.

2.1 When to Review

A good time to start thinking about the Post Implementation Review is when members of the project team remember the most – shortly after the project has been delivered, and when most of the problems have been ironed-out. Start to list ideas and observations while they are still fresh in people's minds. However, to adequately assess the quality of the implementation and complete this process, you'll need to wait long enough for the changes caused by the project to truly take effect. There will probably be a period of adjustment before you can finally review the solution as it was intended to operate: you'll likely need to overcome some of the usual resistance to change, hold people's hands while they operate new systems, and eliminate technical problems that didn't emerge when deliverables were tested. You should therefore typically allow a few weeks, or even a few months, before doing the full PIR. Where possible, allow for at least one, full, successful cycle of business before reviewing lessons learned.

2.2 What to Review

Here are some tips for conducting the PIR:

- **Ask for openness** – Emphasize the importance of being open and honest in your assessment, and make sure that people aren't in any way punished for being open.
- **Be objective** – Describe what has happened in objective terms, and then focus on improvements.

- **Document success** – Document practices and procedures that led to project successes, and make recommendations for applying them to similar future projects.
- **Look with hindsight** – Pay attention to the "unknowns" (now known!) that may have increased implementation risks. Develop a way of looking out for these in future projects.
- **Be future-focused** – Remember, the purpose is to focus on the future, not to assign blame for what happened in the past. This is not the time to focus on any one person or team.
- **Look at both positives and negatives** – Identify positive as well as negative lessons.

When conducting the review, include the following activities:

- **Conduct a gap analysis.**
 - Review the project charter to evaluate how closely the project results match the original objectives.
 - Review the expected deliverables (including documentation) and ensure either that these have been delivered to an acceptable level of quality, or that an acceptable substitute is in place.
 - If there are gaps, how will these be closed?
- **Determine whether the project goals were achieved.**
 - Is the deliverable functioning as expected?
 - Are error rates low enough, and is it fit for purpose?
 - Is it functioning well, and in a way that will adjust smoothly to future operating demands?
 - Are users adequately trained and supported? And are there sufficiently enough confident, skilled people in place?
 - Are the necessary controls and systems in place, and are they working properly?
 - What routine activities are needed to support the project's success?
 - If there are problems here, how will these be addressed?
 - How does the end result compare with the original project plan, in terms of quality, schedule and budget?
- **Determine the satisfaction of stakeholders.**
 - Were the end users' needs met?
 - Is the project sponsor satisfied?
 - What are the effects on the client or end user?
 - If key individuals aren't satisfied, how should this be addressed?
- **Determine the project's costs and benefits.**
 - What were the final costs?
 - What will it cost to operate the solution?

- What will it cost to support the solution in the future?
- How do the costs compare with the benefits achieved?
- If the project hasn't delivered a sufficiently large return, how can this be improved?
- **Identify areas of further development.**
 - Have all of the expected benefits been achieved? If not, what is needed to achieve them?
 - Are there opportunities for further training and coaching that will maximize results?
 - Could you make further changes, which would deliver even more value?
 - Are there any other additional benefits that can be achieved?
- **Identify lessons learned.**
 - How well were the projects deliverables assessed, and how well were timescales and costs assessed?
 - What went wrong, why did these things go wrong, and how could these problems be avoided next time?
 - What went well, and needs to be learned from?
- **Report findings and recommendations.**
 - What have you learned from this review?
 - Do you need corrective activity to get the benefits you want?
 - What lessons have you learned that need to be carried forward to future projects?
 - Does this project naturally lead on to future projects, which will build on the success and benefits already achieved?

2.3 How to Review

As you perform the post-implementation review, certain methods and practices will help you obtain the best possible information:

- **Define the scope of the review beforehand** -The last thing you want to do is to create a political problem. Given the number of people often involved in a project, it's easy to hurt someone's feelings when reviewing the project's success. Clarify your objectives for the review, and make your intentions clear – this will better ensure that people share their experiences openly and honestly. Then make absolutely sure that you stick to these intentions, and that people's egos aren't unnecessarily bruised by the process!
- **Review key documents** – Gather together the key project documents. This will help you assess the project planning process, as well as the actual benefits achieved through the project.

- **Consider using independent reviewers** – Where possible, use outside people in your review process to get an objective, unclouded view of the project. Some people recommend using only independent people in the review, however, you can learn a lot from the perspectives of those who were directly involved in the project – this is why the best strategy is probably to have a balance.
- **Use appropriate data collection** – Collect information in the most appropriate way, for example, by using interviews and surveys. Also, test the deliverable yourself, to make sure you get firsthand information.
- **Deliver appropriate reports** – Report your findings, and publicize the results. Remember that the PIR is designed to help project managers conduct more effective projects in the future, as well as to measure and optimize the benefits of the specific project being reviewed.
- **Present recommendations** – Present the detailed recommendations to the organization and the project leaders, as well as to customers and other stakeholders. Include as many people as necessary so that you keep – and apply – the best-practice information in the future.

As you plan your PIR, be aware of the costs and benefits of the review process itself. Interviewing stakeholders and customers, testing the solution, and documenting the results are time-consuming activities. Make sure the time and resources dedicated to the review are consistent with the project scope and its output, and that the potential benefits of conducting the review are worth the effort put in.

3. Supports Activities

3.1 User training and assistance

- Current employees are trained when the new system is introduced
- New employees typically are trained by user departments, rather than IS staff
- If significant changes take place, the IS group might develop a user training package
 - Special Help via e-mail or company intranet
 - Revisions to the user guide
 - Training manual supplements
 - Formal training sessions

3.2 Information centers

An information center has three main objectives

- To help people use system resources more effectively
- To provide answers to technical or operational questions

- To make users more productive by teaching them how to meet their own information needs

4. System Maintenance

Operations and Maintenance is the fourth phase of the SDLC. In this phase, systems are in place and operating, enhancements and/or modifications to the system are developed and tested, and hardware and/or software is added or replaced. The system is monitored for continued performance in accordance with security requirements and needed system modifications are incorporated. The operational system is periodically assessed to determine how the system can be made more effective, secure, and efficient. Operations continue as long as the system can be effectively adapted to respond to an organization's needs while maintaining an agreed-upon risk level. When necessary modifications or changes are identified, the system may reenter a previous phase of the SDLC.

The Key activities for this phase include:

- Conduct an operational readiness review
- Manage the configuration of the system
- Institute processes and procedures for assured operations and continuous monitoring of the information system's security controls
- Perform reauthorization as required

5. Conclusion

During the life of the system, the Customer support, Maintenance and feedback Phase is the longest and most expensive as the information system provides the most value to the organization in this phase. After system functionally becomes obsolete, the information system is ready to move to retirement in its final phase, the Disposition Phase.

Chapter 6 Conclusion and Future Works Strategies

1. Overview

In the introduction of this report, we expressed the hope that the work in this thesis could be a first step towards a general purpose software system that could be used by the MAMWE company in order to successfully achieve their objective. We also expressed that this work will be of great achievements toward a solution to the problems this company is facing now-a-days. In this chapter we will conclude by describing the progress made towards this goal in terms of our development of a variety of functions that are the core features of the software. Also we will suggest some future improvements so that the system can be as much stronger as possible.

2. Conclusion

The aim of this thesis work has been to develop a web application software for solving the different problems of the MAMWE company. Yet, throughout this report we demonstrated how this software can be implemented and how it can be used for the soul objective. The importance of such a system is of great importance. In addition our work add several advantages to the achievement of the MAMWE objectives. Through a conceptual analysis, we could find the drawbacks of the existing systems. Then with the help of this analysis we have designed and implemented a system which is much capable of handling the MAMWE activities.

The development was straightforward and it was not that much complex. In the first stage we have developed the software template based on the specifications of the company. With the basis of this template we could implement the top common features starting from the system configuration and information input to the development of report based tools dedicated to manipulate the database data to generate meaningful and useful reports. We also introduced some methodologies for organizing data and information in a very convenient and user friendly manner. Security was also a very important issue that we have worked on and implemented it in the software.

3. Suggestions for future works

A good number of problems have to be solved in order to make the system more robust and convenient including :

3.1 Hybrid Profile

This feature is part of the system security. Its main objective is to allow a user to log on two accounts at the same time. This facility may be provided in any user profile given a strong authentication system. The main advantage of hybrid profile is to prevent the unavailability of any task due to several conditions(hardware failure, employee sickness, etc...). In this case other employee can have access to that specific task.

3.2 Online Service

This feature aims to develop a customer online service where they can communicate in real time with the company`s staff for different purposes such as seeking a customer support or asking questions for getting any information.

3.3 Remote Voltage Monitoring

This is a very important feature now-a-days for a power supply company. This system is now existing in the MAMWE company but with huge limitations. The purpose of the development of a similar but more flexible tool is necessary.

References

- [01] Software Requirements Engineering By Linda Westfall
- [02] Pyzdek, T. 2000. *Quality engineering handbook*. New York: Marcel Dekker.
- [03] Wiegers, K. E. 2003. *Software requirements*, 2nd Edition. Redmond, Wash.: Microsoft Press.
- [04] Unified Modeling Language – User guide Grrady Booch, James Runmaugh
- [05] <http://www.ibm.com/developerworks/rational/library/content/RationalEdge/sep04/bell/>
- [06] <http://www.agilemodeling.com/artifacts/classDiagram.htm>
- [07] <http://www.classicsys.com/css06/cfm/article.cfm?articleid=20>
James Hobart
- [08] http://www.ethicsandtechnology.eu/images/uploads/Organizational_structure_and_responsibility
- [09] <http://searchsqlserver.techtarget.com/definition/schema>
- [10] <http://cic.vtt.fi/projects/ifcsvr/memo/vtt-memo-ada-02.pdf>
- [11] <http://www.ciobriefings.com/Publications/WhitePapers/DesigningtheStarSchemaDatabase/tabid/101/Default.aspx>
- [12] <http://publib.boulder.ibm.com/infocenter/rbhelp/v6r3/index.jsp>