

Perpustakaan SKTM

INVENTORY MANAGEMENT SYSTEM FOR DISTRIBUTED CHAIN STORES

NAME : MELISA SUNDRAM

MATRIKS NO : WEK 000129

SUPERVISOR : PN. AZWINA

MODERATOR : MR. CHIEW THIAM KIAN

Abstract

Inventory Management System is an inventory management and organization tool for distributed chains stores that is stores at different locations under the same management. The management at the main office, supervisors and clerks at the chain stores are the main target of this system.

The main purpose of IMS is to allow the main office to monitor the activities of the chain stores and to enable the management to make better business decisions and minimize risks based on the sales of its stocks at the chain stores. IMS would also enhance better communication and interaction between the chain stores and the management at the main office.

The development model used to develop this system is the Waterfall with Prototyping model. This main reason for choosing this model is that the processes in the model can be tailored to meet the specific requirements but changes can still be made to the applications.

The tools used to develop IMS comprise Visual Basic 6.0 and Microsoft SQL Server 2000 is chosen as Database Management System. Crystal Report is used as the report generating tool for the IMS system.

Acknowledgments

It would not have been possible to produce this thesis without the contributions of the following parties in one way or another. Therefore I would like to take this opportunity to express my profound gratitude to the following parties.

Firstly, I would like to thank my supervisor, Pn. Azwina for her invaluable guidance throughout the development of IMS. Not forgetting my moderator, Mr. Chiew Thiam Kian, together their advice and comments have allowed me to be more analytic and discerning when developing the IMS.

Secondly, I would like to thank the staff of MPH Bookstores and 7 – 11 for allowing me to conduct informal interviews. Their kind cooperation enabled me to gather precious information for the development of the system.

Next, I would like to thank my course mates who contributed their knowledge and experience, allowing us to discuss the weaknesses and strength of each other's system. This enabled the improvement of the quality and attributes of the system.

Lastly, I would like to thank my family for giving me the support and encouragement during the development of the system.

Once again I would like to express my gratitude to all the parties mentioned above. Their contribution is very much appreciated. To those that I missed out, I would like to apologize and your kindness will always be remembered.

Thank you.

Table of Contents

Content	Page
Abstract	i
Acknowledgment	ii
Table of Contents	iii
List of Figures	viii
 <u>Chapter 1 : Introduction</u>	
1.1 Project Definition	1
1.2 Project Aim	2
1.3 Project Objectives	2
1.4 Project Scope	3
1.5 Assumptions	4
1.6 Limitations	4
1.7 Report Layout	4
1.8 Project Schedule	7
 <u>Chapter 2 : Literature Review</u>	
2.1 What is Literature Review?	8
2.2 Client Server Architecture	
2.2.1 What is Client Server Architecture?	8
2.2.2 2-Tiered Client – Server Architecture	9
2.2.3 3-Tiered Client – Server Architecture	10

2.3 Operating Systems	
2.3.1 What is an Operating System?	11
2.3.2 UNIX Operating System	12
2.3.2.1 Advantages of UNIX	13
2.3.2.2 Disadvantages of UNIX	13
2.3.3 Microsoft Windows 2000 Professional	14
2.3.3.1 Advantages of Microsoft Windows 2000 P	14
2.3.3.2 Disadvantages of Microsoft Windows 2000 P	15
2.3.4 Linux Operating System	15
2.3.4.1 Advantages of Linux	15
2.3.4.2 Disadvantages of Linux	16
2.4 Database Management Systems	
2.4.1 What is a Database?	16
2.4.2 Microsoft SQL Server 2000	16
2.4.2.1 Advantages of Microsoft SQL Server 2000	17
2.4.2.2 Disadvantages of Microsoft SQL Server 2000	17
2.4.3 Oracle 8i	18
2.4.3.1 Advantages of Oracle 8i	18
2.4.3.2 Disadvantages of Oracle 8i	18
2.4.4 Lotus Notes Database	19
2.5 Authoring Tools	
2.5.1 Active Server Pages (ASP)	19
2.5.2 Visual Basic 6.0	20

2.5.3 Lotus Notes and Domino Release 5	20
2.6 Programming Languages	21
2.6.1 HTML	21
2.6.2 LotusScript	21
2.6.2.1 Advantages of LotusScript	21
2.6.3 JavaScript	22
2.6.3.1 Advantages of JavaScript	22
2.7 Reports Generation	
2.7.1 Why are reports important?	23
2.7.2 What is Crystal Reports?	24
2.7.2 Benefit of Using Crystal Reports	24
2.8 Adobe Photoshop 6.0	25
2.9 Review On Existing Distributed Chain Stores	
2.9.1 7 – 11 Store	25
2.9.2 MPH Book Store	26
2.10 Review on Existing Inventory Management Systems	
2.10.1 Accurate ID Inventory Management System	27
2.10.2 VideoDex Inventory System	28
2.11 Conclusion	29
<u>Chapter 3 : Methodology</u>	
3.1 Waterfall Model	
3.1.1 What is Waterfall Model?	30
3.1.2 Why Not Waterfall Model?	31
3.2 Prototype Model	

3.2.1 What is Prototype Model?	32
3.2.2 Why Not Prototyping Model?	33
3.3 Waterfall with Prototyping Model	
3.3.1 What is Waterfall with Prototyping Model?	34
3.3.2 Advantages of Waterfall with Prototyping Model	34
3.3.3 Stages of the Waterfall Model with Prototyping	36
3.4 Conclusion	37

Chapter 4 : Systems Analysis

4.1 Functional Requirements	38
4.1.1 User Access Authentication	38
4.1.2 Chain Store Information	38
4.1.3 Generating Reports	39
4.1.4 Stocks	39
4.1.5 Receipts and Sales	40
4.1.6 Stock Type Maintenance	41
4.1.7 Notification on Stock Type Shortage	41
4.1.8 Help	41
4.2 Non – Functional Requirements	
4.2.1 Reliability	42
4.2.2 Usability	42
4.2.3 Security	42
4.2.4 Manageability	43

4.2.5 Flexibility	43
4.2.6 On Time	43
4.2.7 Easy to Navigate	43
4.3 Chosen Development Tools	
4.3.1 Operating System	44
4.3.2 Database Management System	44
4.3.3 Authoring Tools	44
4.3.4 Other Programming Tools	44
4.4 Hardware Requirements	45
4.5 Information Gathering	
4.5.1 Internet	45
4.5.2 Books and Journals	46
4.5.3 Informal Interviews	46
4.5.4 Discussions with Lecturer	46
4.6 Conclusion	46
<u>Chapter 5 : Systems Design</u>	
5.1 Process modeling	47
5.2 Data Flow Diagrams	
5.2.1 Context Diagram	48
5.2.2 Diagram 0	49
5.2.3 Diagram 1 for User Access Authentication	50
5.2.4 Diagram 1 for Chain Stores Profiles	50
5.2.5 Diagram 1 for Stock Type	51

5.2.6 Diagram 1 for Stocks	51
5.2.7 Diagram 1 for Help	52
5.2.8 Diagram 1 for Reports	52
5.2.9 Diagram 1 for Shortage Notification	53
5.2.10 Diagram 1 for Receipts	53
5.3 Database Design	54
5.3.1 Entity – Relationship Diagram	54
5.3.2 Data Dictionary	55
5.4 Interface Design	57
5.5 Conclusion	59
<u>Chapter 6 System Implementation</u>	
6.1 Enhancements and Changes to the System	60
6.1.1 Enhancements on the Modules	60
6.1.2 Changes on the Programming Language	60
6.1.3 Main purpose of Reports Generation	61
6.2 Development Environment	
6.2.1 Hardware Configuration	61
6.2.2 Software Configuration	61
6.3 Coding of Modules within IMS	
6.3.1 User Access Authentication	62
6.3.2 Chain Store Profile	64
6.3.3 Stock Type	66
6.3.4 Stocks	67

6.3.5 Receipts	70
6.3.6 Shortage Notification	73
6.3.7 Reports	74
6.3.8 Help	79
6.4 Conclusion	80
<u>Chapter 7 System Testing</u>	
7.1 Software Testing on Inventory Management System (IMS)	81
7.2 Unit Testing of Modules in IMS	81
7.2.1 Ad Hoc Testing on IMS	82
7.2.2 White Box Testing on IMS	82
7.2.2.1 Segment Coverage	82
7.2.2.2 Node Testing	82
7.2.2.3 Compound Condition Coverage	83
7.2.2.4 Data Flow Testing	84
7.2.2.5 Basic Path Testing	84
7.2.2.6 Loop Testing	85
7.2.3 Black Box Testing	85
7.2.3.1 Error Guessing	85
7.2.3.2 Domain Testing	85
7.2.3.3 Module Interface Testing	86
7.2.4 White Box Testing Versus Black Box Testing	86
7.2.5 Unit Testing on IMS	87
7.3 Integration Testing	87

7.3.1 Big Bang Integration	88
7.3.2 Top Down Integration	88
7.4 System Testing	89
7.4.1 System Test Consideration	89
7.4.1.1 The Event List	89
7.4.1.2 Specific Scenarios	90
7.4.1.3 Error Message Testing	91
7.4.1.4 Screen Mapping	91
7.4.1.5 Documentation Testing	91
7.4.2 Fundamental Tests (Product Verification Testing)	91
7.4.2.1 Usability	91
7.4.2.2 Reliability	92
7.4.2.3 Installability	92
7.4.2.4 Performance	92
7.5 Test Case for IMS	92
7.6 Conclusion	92
<u>Chapter 8 System Evaluation</u>	
8.1 IMS System Strengths	93
8.1.1 Ease of control and manipulation	93
8.1.2 User – friendly Interfaces	93
8.1.3 Database Transparency	93
8.1.4 Security	94
8.1.5 Maintenance	94

8.2 IMS System Limitations	94
8.2.1 Speed of data retrieval	94
8.2.2 Predefined User Roles	94
8.3 IMS System Enhancements	95
8.3.1 Include Stocks Supplier	95
8.3.2 Integrating the IMS as part of Teaching Module	95
8.3.3 E - Mail module	95
8.4 Problems Encountered	95
8.4.1 Not familiar with the programming language	95
8.4.2 Difficulty in obtaining end user information	96
8.4.3 Abundance of information in the Internet	96
8.5 Knowledge Gained	96
8.5.1 Knowledge on Additional Software Tools	96
8.5.2 Good Graphical User Interface Design	96
8.5.3 Skills in gathering information and fact	97
8.5.4 Experience in Problem Solving	97
8.5.5 Learning to work independently	97
8.5.6 Skills in writing documentation	97
8.6 Conclusion	97
Bibliography	98
URL	99
Appendix :	
Appendix 1 : User Manual	100

Appendix 2 : Unit Tests	114
Appendix 3 : Test Cases	118
Appendix 4 : User Acceptance Test	123
Appendix 5 : Code Samples in the IMS	134

List of Figures

Chapter 1

Figure 1.1 : Gantt Chart showing the start and end for each activity in the project.

Chapter 2

Figure 2.1 : 2 – Tier Client Server Architecture

Figure 2.2 : 3 – Tier Client Server Architecture

Figure 2.3 : Screenshot of Accurate ID System

Figure 2.4 : Screenshot of VideoDex System

Chapter 3

Figure 3.1 : Waterfall Model

Figure 3.2 : Prototyping Model

Figure 3.3 : Waterfall Model with Prototyping

Chapter 5

Figure 5.1 : Common Notations in Data Flow Diagrams

Figure 5.2 : Context Diagram for Inventory Management System

Figure 5.3 : Diagram 0 for Inventory Management System

Figure 5.4: Diagram 1 for User Access Authentication

Figure 5.5: Diagram 1 for Chain Store Profile

Figure 5.6: Diagram 1 for Stock Type

Figure 5.7: Diagram 1 for Stocks

Figure 5.8: Diagram 1 on Help

Figure 5.9: Diagram 1 for Reports

Figure 5.10: Diagram 1 for Shortage Notification

Figure 5.11: Diagram 1 for Receipts

Figure 5.12: Entity Relationship Diagram for Inventory Management System

Figure 5.13 : a _Login Table

Figure 5.14 : a _Login _Module Table

Figure 5.15 : a _Receipt Table

Figure 5.16 : a _Receipt _Details Table

Figure 5.17 : a _Stock Table

Figure 5.18 : a _Stock _Type table

Figure 5.19 : a _Store table

Figure 5.20 Interface Design for Stock Type

Figure 5.21 Interface Design for Stocks

Chapter 6

Figure 6.1 Software Configuration for IMS

Chapter 7

Figure 7.1 Truth Table for Condition Coverage

Figure 7.2 Unit test for Receipt Module

Figure 7.3 Scenario for System Test

CHAPTER 1
INTRODUCTION
University of Malaya

Chapter 1 – Introduction

1.1 Project Definition

Inventory is defined as a detailed, itemized list or a record of things in one's possession. It included a periodic survey of all goods and materials in stock (*Webster's Revised Unabridged Dictionary, 1996*).

Management is described as the act or practice of managing, handling, supervision or control (*The American Heritage® Dictionary of the English Language, 1998*).

Therefore inventory management can be categorized as the act of directing or controlling items involving quantity of goods and materials at hand. It includes processing, categorizing, checking and recording goods on hand such as groceries, hardware or other materials.

Distributed chain stores give the meaning to stores in different locations under a similar management or control centre. Examples include Malaysian Publishing House (MPH), Guardian Pharmacy and Watson's Personal Care Centre.

The term distributed can be likened to 'client – server' which means splitting an application into different tasks and putting each particular task onto a platform where it can be handled most efficiently. A client is simply a computer with software for a particular service on it that can make requests for information from a server. A server is a computer where the information is stored. It runs server software that understands how to fulfil request for information required by the clients.

Basically, client / server computing is a software architecture that enables distributed computing resources on a network to share common resources among groups users (in this case, the chain stores) at workstations.

1.2 Project Aim

This project aims to develop and enhance the electronic inventory management of distributed chain stores. This is due to the fact that most of the current inventory management applications that are developed are unable to support and suit the specific requirements of the distributed chain stores. Therefore, it is vital to develop a system that is able to suit the needs and wants of the main office and chain stores so that they will be able to function more resourcefully and efficiently.

1.3 Project Objectives

The objectives of the project are as follows:

- a) To design and develop an effective and efficient inventory management system that is valuable to the company.
- b) To provide the main office with effective and efficient information for making business decisions.
- c) To enable better communication between the management at the main office and its staff at the chain stores.
- d) To increase the accuracy and quality in data keeping of goods, records and other relevant documents.
- e) To reduce the cost and time in the conventional paper inventory management system.
- f) To provide the company with a competitive edge in the business world with this inventory management system.

- g) To improve the overall management skills within the company especially in handling and maintaining of data.

1.4 Project Scope

There were several considerations to be made during the course of developing this project. This system would focus on the management at the main office and also the staff working at the chain stores. This system is developed for:

- a) There are two main modules in the Inventory Management System, which are the Main Office Module and Chain Store Module.
- b) The scope extends to the staff at the chain stores. This system will ensure that they are connected to the main management regularly, thus enhancing the operations and management within the store itself.
- c) The scope includes the management at the main office. This system would enable them to monitor and record the activities at their chain stores, increasing better communication between them and the chain stores.
- d) This inventory system is focused only on enhancing the communication between the main office and chain stores, not including the supplier.
- e) The chain stores do not operate 24 hours a day.
- f) The keyboard will assume the role of a scanner as an input device.

1.5 Assumptions

Several assumptions have been made in order to develop this system. They include the following:

- a) Management and staff at the chain stores have basic knowledge in operating and handling a computer.
- b) The staff have fundamental skills on inventory management meaning they do know how to request for goods regardless on paper or electronically.

1.6 Limitations

This Inventory Management System has a number of limitations that are as follows:

- a) This system is meant only for distributed chain stores connected through a network to its main office. It does not include inventory management for a local and single management store.
- b) This system is meant to only handle the inventory of the stores and not other management related issues like company documents and other relevant papers.

1.7 Report Layout

Chapter 1- Introduction

This chapter gives the reader an overall insight on the proposed system. It focuses on the definition and objective of the project as well defining the application.

Chapter 2 – Literature Overview

This chapter studies the various problems and researches done prior to the implementation of the project. It covers through analysis on the domain of the application.

Chapter 3 – Methodology

This chapter discusses the systems properties and architecture as well as the software and hardware intended for the implementation of this application.

Chapter 4 – Systems Analysis

This chapter is focused on the requirements of the application, specifically functional requirements, non-functional requirements, hardware and software requirements.

Chapter 5 – Systems Design

Systems design comprises of combining the application into a whole that contains the functions executed by the application. It covers the user interface design, Data Flow Diagrams (DFD) and modules within the application.

Chapter 6 – System Implementation

The implementation of the application will be thoroughly explained in this chapter. System implementation describes the environment, tools, coding and the development of the individual modules.

Chapter 7 – System Testing

System testing covers the techniques and methods of testing the complete application. All system must go through a series of testing before it is deployed as a fully functional application

Chapter 8 – System Evaluation

System evaluation is the final chapter whereby the entire application is evaluated. This chapter outlines the strengths and weakness of the IMS system. Suggestions for future enhancements of this application, problems faced and experience gained throughout the project will be included.

Conclusion

This is the conclusion of the whole report.

[illegible]

Chapter 2 – Literature Review

2.1 What is Literature Review?

Literature Review is a fundamental process when developing a system. It consists of thorough planning and research to ensure the success of the system developed. This is to make certain that the objectives of this system are achieved and to gain a proper understanding of the system.

In the case of inventory management for distributed chain stores, research was conducted in related areas of the system, including client-server architecture, programming languages used and other software to enhance the Inventory Management System.

2.2 Client Server Architecture

2.2.1 What is Client Server Architecture?

Client – server architecture basically means splitting an application into tasks or jobs and placing each of them onto a platform where they can be handled most effectively.

A client can be a computer with client software, making a request for information from a server. A client is usually responsible for presenting information to the user. A server is a computer where the information requested is stored. The server contains server software that would then retrieve and return the data to the client. Applications must be split carefully between the client and the server in order to optimise the response time of the system (*Rand Dixon, 1995*).

Client – server can also be viewed as a network environment where the data is maintained, handled and controlled by the server and is available for access by the client. The clients are

generally single user computers or workstations that provide high user-friendly interface to the end user. The server provides shared services to the clients. The most common type of server is the database server. The server enables many clients to share access to the same database and manage the database at a high performance level.

There may be one centralised server or several distributed servers. This allows clients and server to be placed independently at different locations on the same network, possibly on different hardware and operating systems.

There are 2 types of client – server architecture, namely 2-tiered and 3-tiered client server architecture.

2.2.2 2-Tiered Client – Server Architecture

A 2-tiered client – server architecture involves only 2 computers: a client and the server. The 3 components within the architecture that is application – presentation, data and processing are divided between these 2 computers. Presentation is handled by the client and the data is stored and accessed usually through a database server. Processing is maintained between the client and the server.

The client computer handles the application logic and the database server concentrates on handling data intensive tasks. The clients would request for information from the server using Structured Query Language (SQL). However in order for the client to send the SQL, it first has to know where the data resides, how the physical data looks like and the syntax of the server. The server would then return the data requested back to client.

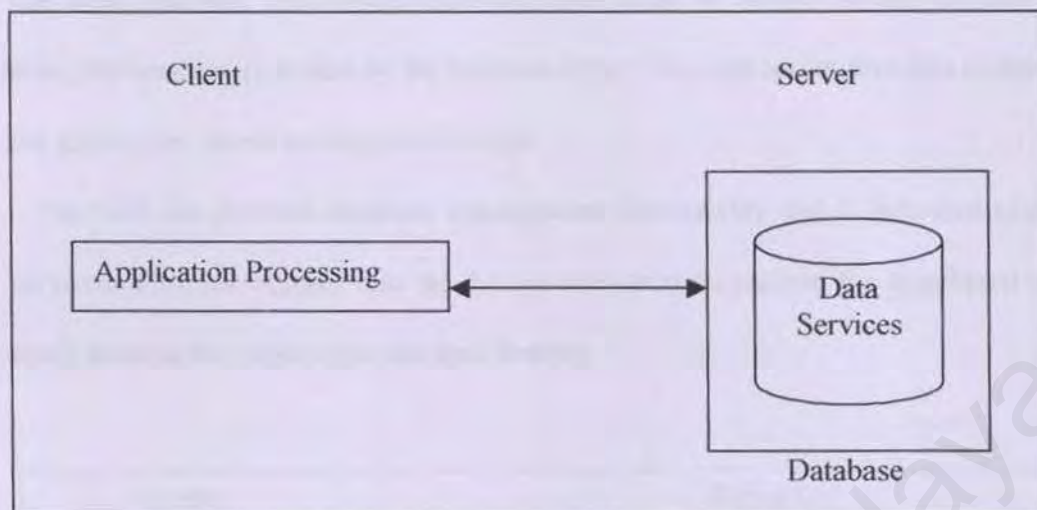


Figure 2.1 2 – Tier Client Server Architecture

2.2.3 3 – Tiered Client Server Architecture

3-tier client architecture is divided into 3 distinctive and logically separate tiers. These tiers are the presentation tier, functionality tier and data tier.

The top layer, known also as the user systems layer contains user services such as session, text input, dialog and display management.

The middle tier provides process management services, i.e. process monitoring, enactment and resource that may be shared by multiple applications. It also controls transactions and queuing to ensure proper completion of transaction. The middle tier can be divided into 2 or more units with different functions. This is usually implemented for some Internet applications. These applications contain light clients usually written in programming languages like HTML and applications server written in C++ or Java. Usually the gap between these 2 layers is too wide. Therefore an intermediate layer, a web server is needed to

link them together. This layer receives request from an Internet client and generate HTML using the services provided by the business layer. The web server provides isolation between the application layout and application logic.

The third tier provides database management functionality and is dedicated to data and file services. This tier ensures that the data is consistent throughout the distributed environment using features like replication and data locking.

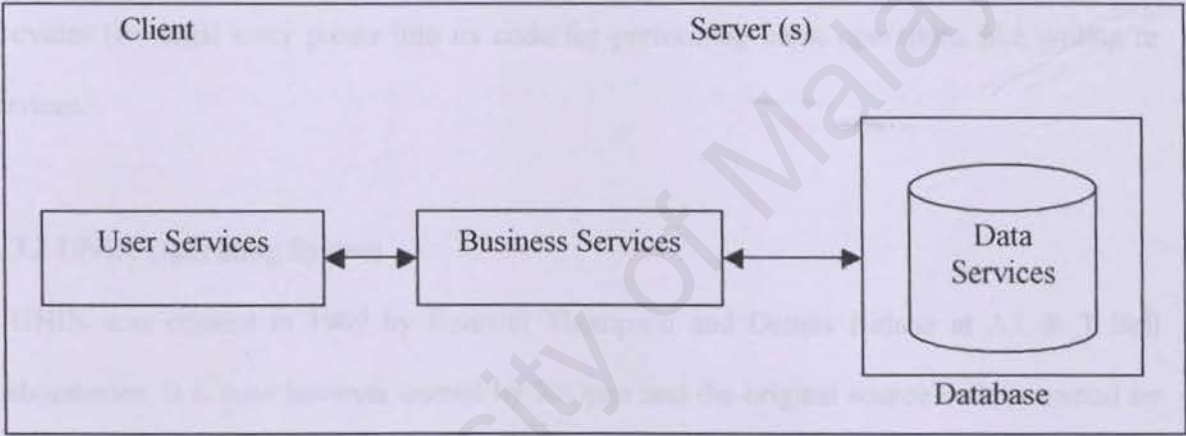


Figure 2.2 3 – Tier Client Server Architecture

2.3 Operating Systems

2.3.1 What is an Operating System?

An operating system is a layer of software that takes care of technical aspects of a computer's operation. It shields the user of the machine from the low-level details of the machine's operation and provides frequently needed facilities.

Normally the operating system has a number of key elements: (i) a technical layer of software for driving the hardware of the computer, like disk drives, the keyboard and the screen; (ii) a file system which provides a way of organizing files logically, and (iii) a simple command language which enables users to run their own programs and to manipulate their files in a simple way. Some operating systems also provide text editors, compilers, debuggers and a variety of other tools. Since the operating system (OS) is in charge of a computer, all requests to use its resources and devices need to go through the OS. An OS therefore provides (iv) legal entry points into its code for performing basic operations like writing to devices.

2.3.2 UNIX Operating System

UNIX was created in 1969 by Kenneth Thompson and Dennis Ritchie at AT & T Bell Laboratories. It is now however owned by X/Open and the original source code is owned by The Santa Cruz Operation (SCO).

Besides the traditional operating systems components, it includes a set of libraries, applications, file and process control. It is an ideal platform for running mail servers and network file systems at a low cost.

UNIX can be used for storing files that include user personal files as well as publicly accessible software archives. It also manages centralized databases and serving information to remote users.

It is capable of running a web server and storing web pages. Because of its durability, it can be left on for 24 hours a day. UNIX can implement shared network file systems. It also

provides remote services. This means that users can actually request for information without logging in.

2.3.2.1 Advantages of UNIX

- Excellent connectivity. UNIX, being the operating system of the Internet and World Wide Web, has always had to offer connectivity options.
- With over 25 years in the marketplace, UNIX systems tend to be very reliable. Note however, that because modifications tend to be proprietary to a specific vendor it is possible that modifications do not make it back into the UNIX base that is used by all vendors.
- UNIX has high scalability UNIX technology has run on machines from the original 8086-based PC to multi-million dollar supercomputers.
- Supports multi-user and multi-tasking. UNIX has always been capable of running multiple programs at one time and supporting multiple users simultaneously. This means that file, print and remote access servers can be implemented using any UNIX-based system.

2.3.2.2 Disadvantages of UNIX

UNIX runs on powerful workstations. Therefore it's not cost effective to use especially for smaller organizations.

UNIX is very expensive system to maintain.

2.3.3 Microsoft Windows 2000 Professional Operating System

MS Windows 2000 Professional is arguably the most user – friendly operating system around these days. It can be used to connect to either Internet or Intranet sites, access files and run software applications.

MS Windows 2000 Professional is an upgraded version of Windows NT that consists of the advantages of the Windows NT operating systems and newer benefits included in MS Windows 2000 Professional. It is easy to use and steer and has plenty of icons to assist user in navigating through the system.

2.3.3.1 Advantages of Microsoft Windows 2000 P

- It includes an improved diagnostic tool that contains information on devices, driver information and system resources. This is all presented in a graphical tool that makes it easier to view and examine the system.
- It has a System Policy Editor which enables administrators to manage and maintain user's desktop in a consistent and reliable manner.
- The task manager provides information on each application and processes that run on the computer, making it easier to monitor each activity.
- The network monitor enables network traffic to and from the server to be examined at packet level. This is later used for analysis, thus making troubleshooting network problems easier.

- MS Windows 2000 Professional is able to combine information from several applications into a single compound document using OLE (Object Linking and Embedding) capabilities of windows – based applications.

2.3.3.2 Disadvantages of Microsoft Windows 2000 P

- Requires shutdown and rebooting whenever changes occur to the operating system. Rebooting is needed when gateway address or computer name is changed.
- High maintenance and support requirement make them costly to run.

2.3.4 Linux Operating System

Linux was developed by Linus Torvalds. It is basically another version of UNIX operating systems. Therefore, it shares the same UNIX commands.

It is a multitasking system that allows users to run many programs on the same system at once. It was built to support most common Internet protocols, including Electronic Mail, Usenet News, Gopher, Telnet, Web, FTP, Talk, POP, NTP, IRC, NFS, DNS, NIS, SNMP, Kerberos, WAIS and many more.

2.3.4.1 Advantages of Linux

- Linux is used widely on web servers because of its high fault-tolerance. It is resistant toward most viruses on the net.
- It has a lower cost when compared with MS Windows NT and UNIX operating system since Linux is available for free via the net.

- It is cost – effective. Linux has the ability to scale the size of the site as traffic grows.
- Relatively easy implementation and maintenance as source codes are freely available to everyone.

2.3.4.2 Disadvantages of Linux

- Linux is quite insecure since its source code is available on the net, making it prone to hacking.
- Since it is developed worldwide, there is a lack of proper organized support and documentation.

2.4 Database Management Systems

2.4.1 What is a Database?

A database is a collection of associated and related data that is shared by different categories of users. In order to maintain the data and to reduce redundancy, a database management system (DBMS) is used. A DBMS is software that allows the user to define, create, update and arrange data in a database. It allows an organization to manage the data easily and optimally. It acts as an intermediary between the users and the database itself.

2.4.2 Microsoft SQL Server 2000

Microsoft SQL Server 2000 is a relational database system for getting information and updating a database. Queries take the form of a command language that allows users to select, insert and update the data in the DBMS.

2.4.2.1 Advantages of Microsoft SQL Server 2000

- Microsoft SQL Server 2000 provides online backups and enhanced failure clustering. This means that it allows data to be backed up while the database is still online and accessed by users. Microsoft SQL Server 2000 also includes the capability to perform differential backups for differential database.
- It has important and significant security enhancements. It can be installed with a much higher level of security, with the advantage of integrated security provided by Microsoft Windows 2000 Professional.
- MS SQL Server 2000 is highly reliable and scalable, which is vital for the success of an enterprise database.
- This DBMS provides efficient support for text and images. Although they aren't stored within a data row, they are stored in a separate collection of pages of their own.
- Multiple instances of Microsoft SQL Server 2000 allows greater flexibility to the Applications Service Provider (ASP), thus allowing independent installations of Microsoft SQL Server 2000 on the same machine. If an application fails, it only affects itself and not other applications running on other instances of Microsoft SQL Server 2000. This indirectly enables users to test different versions of an application on a single server.

2.4.2.2 Disadvantages of Microsoft SQL Server 2000

- Microsoft SQL Server 2000 runs only under Windows, therefore the Operating Systems must be a Microsoft OS i.e. MS Windows NT or MS Windows 2000 Professional.

2.4.3 Oracle 8i

Oracle 8i is a Database Management Systems which main targets include high – end workstations and minicomputers as the server platforms in which it runs itself. Amongst the products introduced by Oracle are enterprise business applications and decision support tools.

2.4.3.1 Advantages of Oracle 8i

- Oracle is a very large database, able to support extensive data, up to terabytes in size.
- It can support many users simultaneously executing database applications on the same data. It minimizes data conflict and maximizes data concurrency.
- Oracle has a high transmission rate which guarantees faster processing performance.
- Oracle has a highly available, working up to 24 hours straight a day with no downtime.
- It provides file-safe security features that are able to limit and monitor data access.
- Oracle is able to combine data physically located on different computers into one logical database that can be accessed by users on a network.
- It also allows different types of computers and operating systems to share information across a network.

2.4.3.2 Disadvantages of Oracle 8i

- The cost of an Oracle database site license is very expensive, up to a few thousand dollars.
- Oracle also requires experienced database administrators who must perform continual maintenance to ensure optimal performance of the database.

2.4.4 Lotus Notes Database

Lotus Notes applications are called database, at times giving the impression that it is similar to other DBMS such as MS Access, Oracle and MS SQL Server 2000. However, it is important to note that Lotus Notes is not a Relational – DBMS whereby in a typical Relational – DBMS, all the information are kept structured and organised in tables.

Data in Lotus Notes may be kept in columns which can be likened to the tables in a Relational – DBMS, whereby each column hold a field which contains specific information such as Date, Name and Item ID.

Information can still be queried but this time instead of the conventional way using tables in an R – DBMS, columns are used to sort out the information required.

2.5 Authoring Tools

2.5.1 Active Server Pages (ASP)

Microsoft Active Server Pages (ASP) is the server – side execution environment in the Internet Information Server (IIS) that runs ActiveXTM and ActiveX server components on the server. With ASP, websites with creative and powerful content are built easily, incorporating languages like Hypertext Mark Up Language (HTML) along with scripts such as VBScript and JavaScript.

ASP allows websites to be customised according to the users request. Websites may be linked to other websites or to a database so that users get the latest information.

ASP is easy to learn if there is basic knowledge on HTML and relatively easy to implement. It supports other programming languages such as VBScript, JavaScript and

Python. ASP has a large user-support group, making it easy to download codes and participate in discussions.

2.5.2 Visual Basic 6.0

Visual Basic is a high level, object oriented programming language evolved from the earlier DOS version called BASIC. In Visual Basic, programming is done in a graphical environment. Visual Basic is made up of many subprograms, each has its own program codes and can be executed independently and at the same time each can be linked together in one way or another.

The structure of the Visual Basic programming language is very simple, particularly as to the executable code. Its most notable feature is its graphical user interface that provides appealing views for the management of the program structure in the large and the various types of entities such as classes, modules and forms.

2.5.3 Lotus Notes and Domino Release 5

Lotus Notes and Domino are a set of programs designed to enable groups of people to work together efficiently. (*Dorothy Burke and Jane Calabria, 2000*). It is a database-driven groupware application that manages information for many users on the network to communicate, collect, and share database documents following a client/server model. Lotus Notes enables groups of people to work together regardless of technical, industrial or physical boundaries. Among its notable functions are organising schedules and group calendars. It is designed to handle tasks that normally require many applications to complete.

2.6 Programming Languages

2.6.1 HTML

Hyper text Mark Up Language (HTML) is a language used to construct a web page, which may include elements such as text, graphics and sounds (*Powell & Whitworth, 1998*).

HTML codes are written in standard American Standard Code for Information Interchange (ASCII). The web pages are usually viewed by a variety of browsers, the most popular being MS Internet Explorer or Netscape Navigator. Therefore, HTML is used to request a Web browser to format and display a web page in a specific way. HTML codes usually incorporate codes that create titles, bold texts, and links that appear in the web page.

2.6.2 LotusScript

LotusScript is a cross platform, BASIC object – oriented programming .It allows a large variety of complex scripts which can be placed in various locations and events in Lotus Notes. It provides a common programming environment across Lotus applications on platforms supported by Lotus Notes.

2.6.2.1 Advantages of LotusScript

- LotusScript is a multi-platform BASIC-like scripting language and is not dependable on a particular operating system. This means that scripts developed on Linux would be able to execute unchanged on Windows platform.

- LotusScript is an object – oriented language and contains Object Classes whereby scripts can be written to access and manipulate objects in these classes. These scripts are based on events in Lotus Notes, for example opening a document or clicking an object.
- LotusScript includes Object Linking and Embedding (OLE), linking Lotus Notes to other applications like Microsoft Office and Lotus SmartSuite.
- LotusScript allows users to create their own objects and classes. These objects and classes are called LotusScripts Extensions (LSX). These classes can support functions like inheritance and constructors and destructors. This is an advantage to users who are familiar with object – oriented programming like C++ and Java.

2.6.3 JavaScript

JavaScript language was created by Netscape in 1996 and initially included in Netscape Navigator 2.0 browser via an interpreter that reads and executes the JavaScript included in .html pages. The language has steadily grown in popularity and is now supported by browsers including MS Internet Explorer.

2.6.3.1 Advantages of JavaScript

- JavaScript provides interactivity for web pages without relying on server – side Common Gateway Interface (CGI) programming, allowing the web pages to be interactive even when there is no connection to the Internet.

- It contains Java applets that produce animated images that enhances the application, making it attractive and interesting to explore.
- JavaScript is able to validate whether information entered into a particular field is relevant to the type of field, without requiring the server program (CGI) to check the field entry and then report back to the user. This therefore saves time and server processing power.

2.7 Reports Generation

2.7.1 Why are reports important?

Perhaps the most important module in the Inventory Management System is the reports generation tool. This is because the transactions and sales that occur at the chain store level can be likened to raw data which consist mostly of numbers and very particular details. This details are then processed into information in the form of reports.

The reports which are then generated would give the administrators at the main office the knowledge they need to make business decisions. Based on the graphs which are generated from the reports, they can make important choices on what particular stock sells better when compared to other stocks.

This in the long run would help them to minimize business risks such as purchasing an item which might not sell very well in a certain chain store. Therefore while minimizing the risks and bad business decisions, the reports can actually indirectly help to increase the overall revenue and profits of the company.

2.7.2 What is Crystal Reports 8.5

Crystal Reports is a data access and report generation and management tool for databases. A report is simply an organized presentation of data. Its purpose is to provide management with the information it need to run its organization efficiently. Therefore, Crystal Reports allows the creation of complete, customized, attractive management reports quickly and easily.

2.7.3 Benefit of Using Crystal Reports

- Crystal Reports is supported by multiple platforms such as MS Windows 2000 Professional, MS Windows NT and UNIX.
- It is able to export reports is various formats, to name a few i.e. PDF, HTML, XML, RTF, Microsoft Word, Microsoft Excel, a variety of text and email formats.
- The Report Engine has increased capacity to process multiple, simultaneous jobs, without compromising the performance of the report when opening, sorting and formatting it.
- The Report Designer interface offers better control over the positioning of report objects. This increases the ability to move, align, resize, copy and paste multiple objects, and the alignment and ruler options.
- The RPT file format creates smaller files with faster decompression speed. This format provided efficient storage and space for complex reports.

- Crystal Reports provides advance grouping and summarizing. It supports hierarchical grouping in which data is arranged in a report to show hierarchical relationships between the data.
- Increased operation performance for SQL database servers especially in faster report processing and better use of network resources.

2.8 Adobe Photoshop 6.0

Adobe Photoshop is a digital graphic application that allows users edit and make changes to pictures and other visual graphics.

The primary unit in Adobe Photoshop is the pixel that is actually a representation of a colour unit that controls the hue, saturation and brightness. Changes are done on the two – dimensional level whereby a paintbrush can be dragged across a picture to replace the original pixels with new ones. This makes the manipulation and editing of existing images easy and unconstrained.

2.9 Review on Existing Distributed Chain Stores

2.9.1 7 – 11 Store

This 7-11 store is located in Section 17, Petaling Jaya, Selangor. It is a retail chain store that specializes in sundry, toiletries and stationery products.

They take stock by checking the items on a regular basis. If they wish to replenish an item, they directly telephone the supplier and state the amount they need. They do not have an

inventory management system that informs them automatically if a particular item is at a critical level and needs to be replenished soon.

At the sales level, they key in the price of an item and not its ID Number, which results in them not knowing what item was sold.

At the end of each month, reports on the daily sales are printed and sent to the main office, as the only line of communication between the main office and the chain store. There is no daily automated communication between the main office and the chain store.

2.9.2 MPH Book Store

MPH bookstore is located at Section 14, Petaling Jaya, Selangor. It is a large bookstore that focuses on books, magazines and newspapers.

MPH does have an adequate Inventory Management System, whereby each book sold is updated immediately on the store's database and is updated the next day in the main office's database.

Any item bought is scanned with a scanner and not keyed in directly, thus reducing the possibility of keying in the wrong ID Number.

Reports are generated daily. Once collected, they are sent weekly to the main office for reference purposes.

2.10 Review on Existing Inventory Management Systems

2.10.1 Accurate ID Inventory Management System

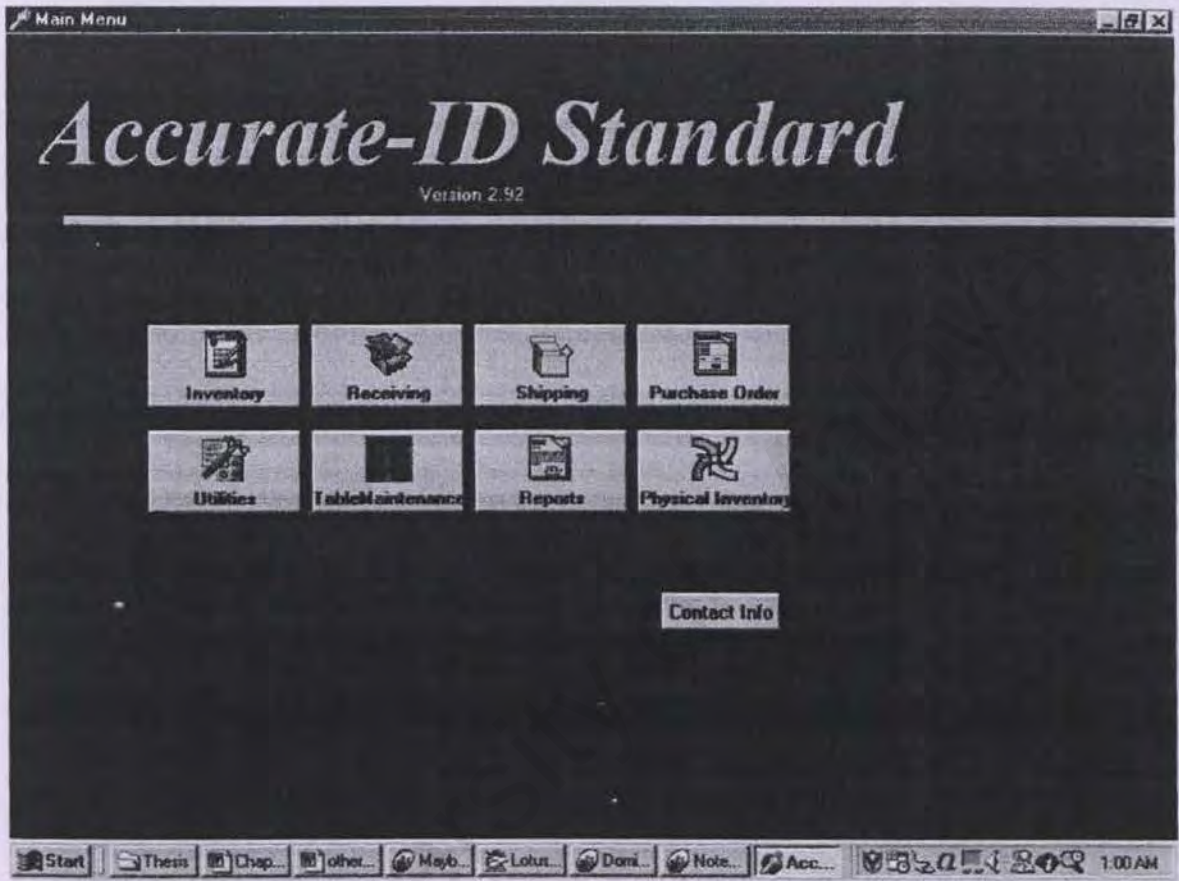


Figure 2.3 Screenshot of Accurate ID System

Accurate ID Inventory Management System in figure 2.3 contains several modules such as receiving and shipping goods and printing reports. It also contains information of the vendors and customers of the store.

Its notable features include its ability to generate reports by goods that were shipped or received, by date, by category and location. The screens contain buttons that makes it easier for even first time users to understand and user the system.

However, this system is limited to single – operation stores and not distributed chain stores. It doesn't show the interaction between chain stores and main management. In this system, users are only able to look up only by item description. This requires the users to remember the items description and limits the flexibility of searching for a particular item such as by ID Number.

2.10.2 VideoDex Inventory System

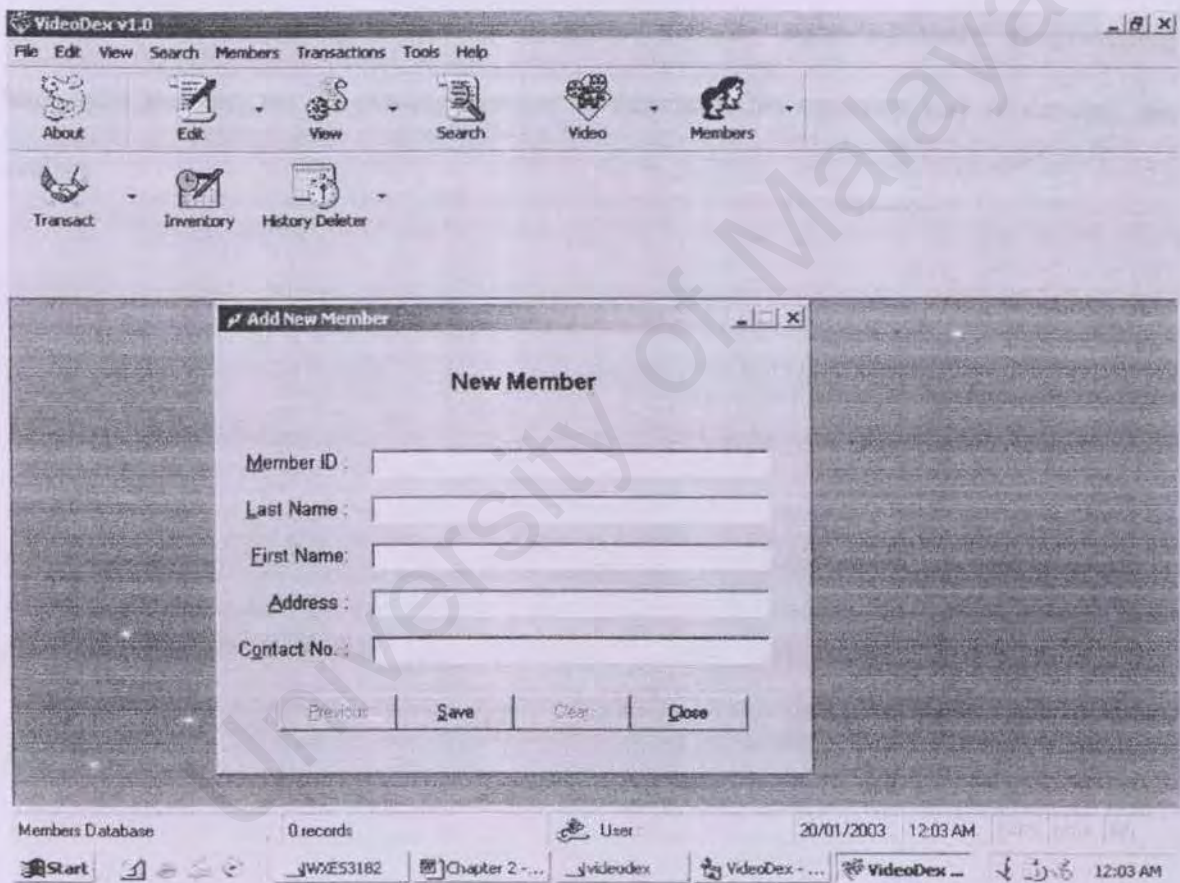


Figure 2.4 Screenshot of VideoDex Inventory System

VideoDex Inventory System manages the inventory of a video rental store. It contains modules such as Members, Transactions and Inventory.

Among the noteworthy features is that this particular inventory system is client – server based which enables it to function in more than one workstation. It has a relatively easy to understand interface and contains a search function which is able to search both the video and member database.

2.12 Conclusion

This chapter consists of intense research and study on the various tools. This includes reading up and comparing server architectures, DBMS, authoring tools, programming languages and reviews on existing systems to determine the optimum way of developing the system.

Chapter 3 – Methodology

3.1 Waterfall Model

3.1.1 What is Waterfall Model?

In this process model, the stages are depicted as cascading from one to another. It specifies that one stage must be finished before proceeding to the next stage. This means once all the requirements have been gathered from the users, thoroughly analysed and documented in a requirements document, then only can the design stage commence. The waterfall model presents a very high – level view of what occurs during the development of the system and also suggests what the developers can expect to encounter (Pfleeger, 1998).

The waterfall model is useful in helping the developers determine what they need to go. Customers, even those who aren't familiar with the system's development find it easier to understand this software life cycle model. This model makes it clear what products are necessary in order to begin the next stage of development. Besides this, the waterfall model is the base in which other software life cycle model were created.

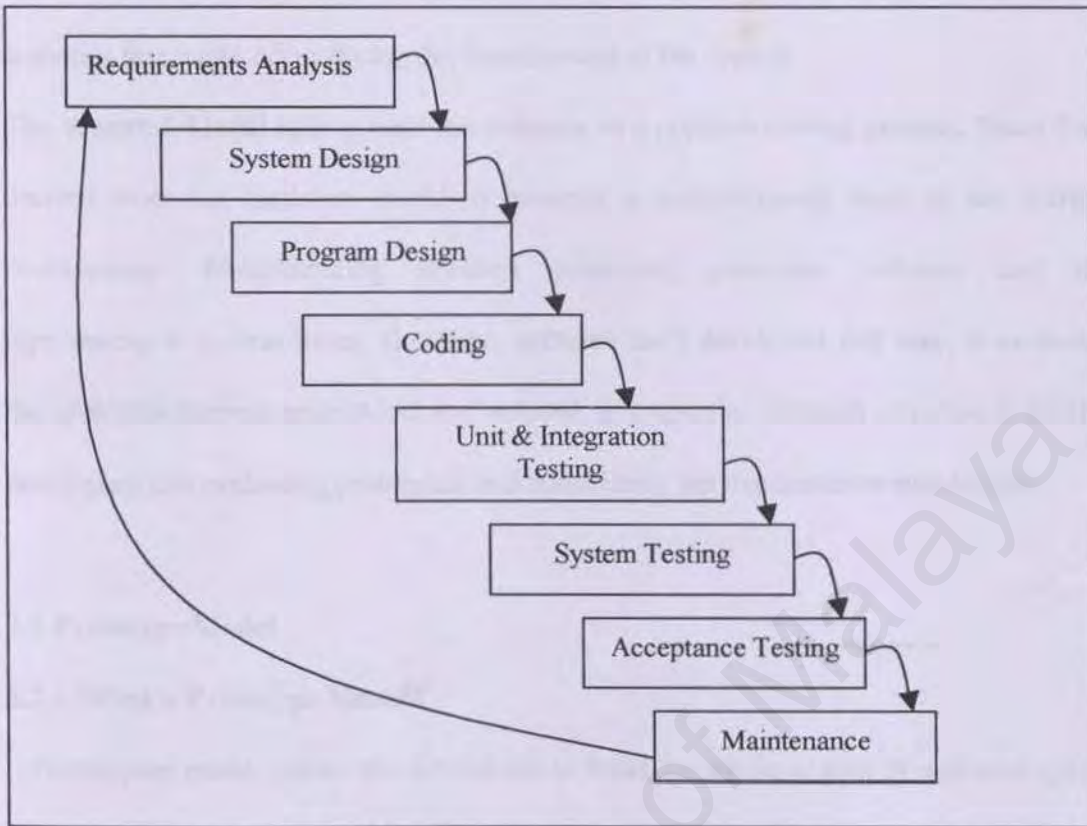


Figure 3.1 Waterfall Model

3.1.2 Why Not Waterfall Model?

One of the main drawbacks with the Waterfall Model is that it does not reflect the way the codes are developed. With the exception of well understood problems, software is usually developed with a great deal of iteration. The software is often used in a solution to a problem that had never been solved. During the actual software development process, developers may trash from one activity to the next and then back again, as they try to gather information about the system.

Besides that, the Waterfall Model shows no insight into how each activity transform from one stage to another, for example from requirements analysis to design. Therefore

the model provides insufficient guidance to developers on how to handle changes to activities that might occur during the development of the system.

The Waterfall Model fails to treat the software as a problem solving process. Since it was derived from the hardware world, it presents a manufacturing view of the software development. Manufacturing involves producing particular software and then reproducing it several times. However, software isn't developed that way, it evolves as the problems become understood and reduced. In particular, creation of software involves developing and evaluating prototypes and considering the requirements and designs.

3.2 Prototype Model

3.2.1 What is Prototype Model?

Prototyping model allows the developers to build the whole or part of a system quickly to understand and clarify issues. The system developer builds a prototype and then lets the user interact and experiment with it. Usually requirements and design are repeatedly examined to ensure that the users, customer and developer have a common ground on the functionality of the system. This model focuses on reducing risk and ambiguity in development.

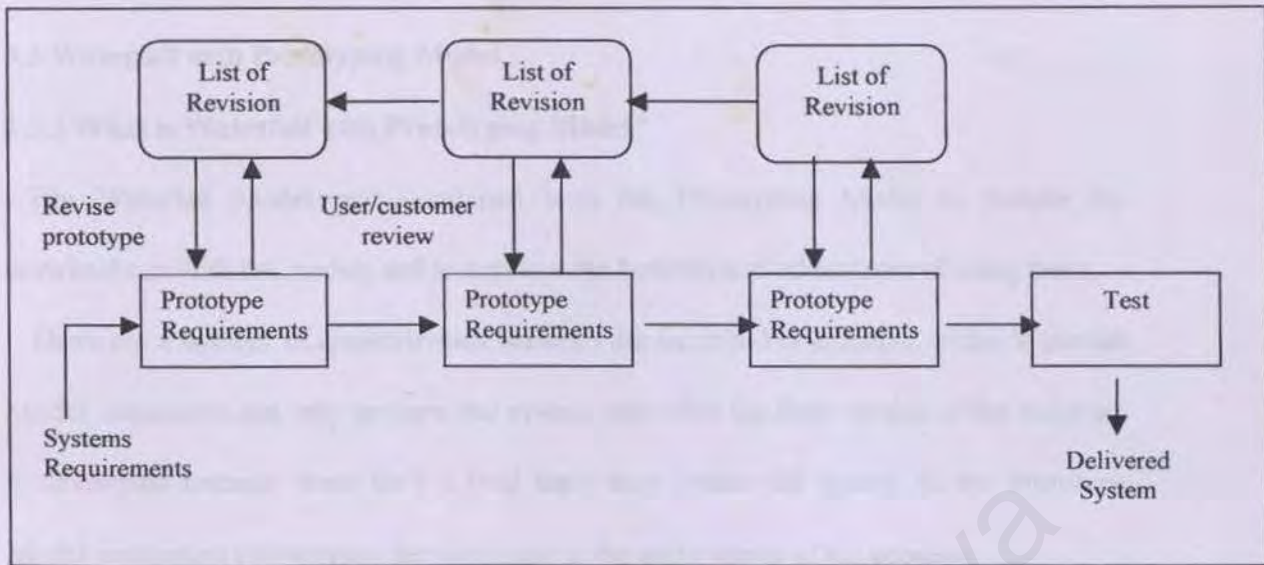


Figure 3.2 Prototyping Model

3.2.2 Why Not Prototyping Model?

When trying to meet the schedule of the whole project, the prototype may be accepted in an unfinished state or without necessary refinements. Although this may seem to cut short the development time and effort, in the end neither the developers nor the users have anything to gain.

The developers do not tend to consider the long – run maintenance of the prototype. They justify this fact by saying that the next release of that particular software is more refined and solves the problems that occurred in the previous prototype. In this case, the manufacturers do not take responsibility for their actions.

The cost to build a prototype might be surprisingly large at times and require a large sum of the total cost of developing the system.

3.3 Waterfall with Prototyping Model

3.3.1 What is Waterfall with Prototyping Model?

The Waterfall Model was combined with the Prototyping Model to reduce the drawbacks in both the models and to improve the benefits and advantages of using them.

There are a number of dissimilarities between the models. For example, in the Waterfall Model, customers can only preview the system only after the final version of the software is developed because there isn't a feed back loop within the model. In the Prototype Model, customers can preview the prototype at the early stages of the process.

When following the Waterfall Model, developers aren't allowed to modify the requirements of a previous stage until the next iteration, as opposed to the Prototype Model, whereby developers can refine or add requirements to the system after the prototype is built.

The complexity of an error tends to increase in the Waterfall Model because each phase is sequential to the other. However in the Prototype Model, the error is usually low because the prototype allows the developer to detect any deficiencies early in the process.

There are few notable similarities between the 2 models. Each lifecycle is divided into phase where specific objectives are met. Both models have an objective to reduce the development and maintenance cost of the system.

3.3.2 Advantages of Waterfall with Prototyping Model

- It is easy to allocate each milestone with its deliverable and lay out tasks that need to be done.
- User involvement in the early stages ensures that the system does not deviated from the user's need.

- This model provides the opportunity to explore alternative strategies and to make revisions on the prototype.
- Emphasizes completion of each phase before moving on.
- Stresses on testing as a fundamental part of the model.
- Early planning, customer input and design are particular importance of the model.

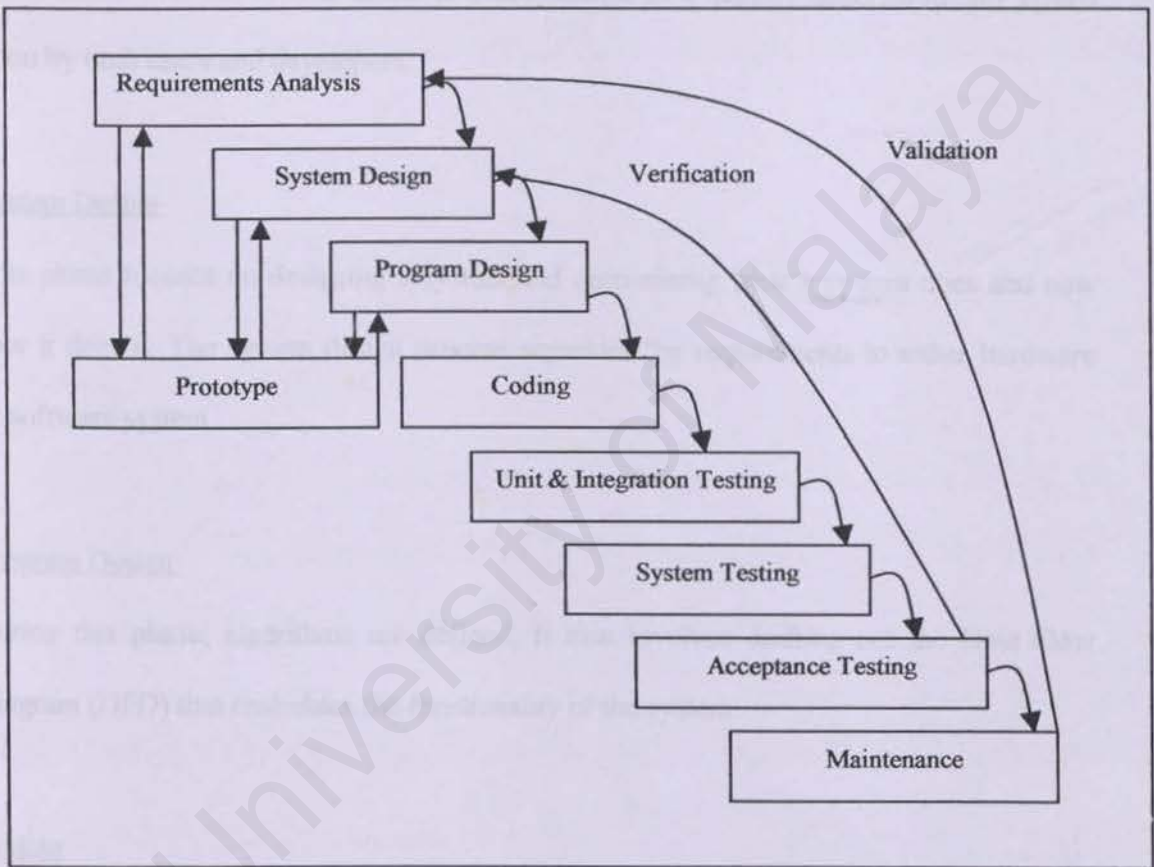


Figure 3.3 Waterfall Model with Prototyping

3.3.3 Stages of the Waterfall Model with Prototyping

Requirements Analysis

This phase involves information gathering through the Internet, interviews and reading materials such as journals, magazines, books and newspapers. Survey and comprehensive research is done. The systems services, constraints and goals are established usually by consultation with the system users. It is defined in a manner understood and agreed upon by both users and developers.

System Design

This phase focuses on designing a system and determining what a system does and how it does it. The system design process separates the requirements to either hardware or software system.

Program Design

During this phase, algorithms are defined. It also involves drafting out the Data Flow Diagram (DFD) that resembles the functionality of the system.

Coding

This phase centres on transforming the algorithms defined during the previous phases into a language understandable by the computer, using programming languages and application development tools based on the design specifications.

Unit Testing

This stage ensures that each module acts accordingly to its specification defined during program design stage. It also checks for any inconsistencies in the modules.

System Testing

This stage determines that the entire system behaves accordingly to the Software Requirements Specification (SRS).

Maintenance

This phase continues the detection and repair of bugs in the system.

3.4 Conclusion

This chapter is mainly on determining the methodology that will be used to develop the system. After thorough research, the model chosen was the Waterfall with Prototyping model for it is able to meet the specific requirements of the system.

Chapter 4 – Systems Analysis

4.1 Functional Requirements

A functional requirement describes how the system interacts with its environment. It captures the tasks that the system has to perform. However, it does not include details on the implementation of the system such as what hardware or software the system must use.

There are 8 modules to be included in the system. They are as follows:

4.1.1 User Access Authentication

The purpose of user access authentication is to determine if the person attempting to log on to the system is a valid user based on their user id and password. This module will decide which user has the access to which module based on their given roles. The administration at the office can have access to all modules, the supervisor can have access to the stock and the report module for the items at his chain store. Clerks have access to receipt module only.

4.1.2 Chain Store Information

This module purpose is to maintain details on each and every chain store that belongs to that particular business. It contains information on the location of the chain store, its supervisor and other relevant details and can only be accessed by the main office.

The administration at the main office can add, delete or edit the chain store profiles.

Below are the fields that are included in Chain Store module :

- i) Chain Store Id
- ii) Chain Store Name
- iii) Address
- iv) Telephone Number

- v) Date Opened
- vi) Supervisor Name

4.1.3 Generating Reports

This is a very important module in the system. The purpose of reports is to enable the administration at the main office at the Decision Support System to make business decisions based on the Transaction Processing System of transaction and sales that occur at the chain stores. The reports generated would give the administration the knowledge they need to make decisions based on which items sell better or which location makes the most profit. Graphs generated by the report would give the administration a visual tool to compare the performance of the chain stores. Reports can be generated automatically by the user from chosen date to another chosen date. The administration at the main office can generate reports on all the chain stores. However, the supervisors of the chain stores can only generate reports based on their respective chain store. Reports generated are based on the

- i) total sales of each chain stores
- ii) The number of stocks left in the chain stores
- iii) The number of stocks that need to be replenished
- iv) The number of a particular stock type (item) sold in all chain stores

4.1.4 Stocks

Stock module contains information on the stock that exists in a chain store. The administrator can select which store to add or remove an existing stock. Both he and

the supervisor can determine the quantity of an item to be placed at the chain store and set the critical quantity. However the supervisor can only edit the existing and critical quantity of items in his chain store only and not other chain stores. He can't add a new item into his chain store because this function is restricted only to the administration at the main office to maintain better control and management of the chain stores. Should a stock quantity be less than the critical quantity, the whole row will be bolded. This makes it easier for the administrator and supervisor to identify which stock need to be replenished.

4.1.5 Receipts and Sales

This module keeps track of the items that sold at the chain store. Its monitor the quantity of the item sold and displays information such as Item ID Number, Description, Quantity, Price and Total Price. It can likened to the sales receipt at the chain store. It is accessed at the chain store level by the sales clerks. However the supervisor and the administrator can view the receipt details on the sales. Again the supervisor can only view the receipts of his own chain store and the administrator has access to all the receipts from all chain stores. This module contains :

- i) Date of purchase
- ii) Sales Id
- iii) Item Id
- iv) Description
- v) Quantity
- vi) Unit Price
- vii) Total Price
- viii) Clerks Name
- ix) Clerks Id

4.1.6 Stock Type Maintenance

This module focuses on the items in the inventory list. The administration at the main office has access to this module in order to add, edit or remove a particular item from the system. The information of the items are as below :

- i) Item Id
- ii) Description
- iii) Quantity
- iv) Weight
- v) Cost Price
- vi) Sales Price

4.1.7 Notification on Stock Type shortage

This notification will display all the stock types that are below the critical quantity. Therefore this would alert the administrator to replenish the stocks that are below the critical amount by clicking the Stock module. The supervisor would get the same notification but is limited to stocks that are below the critical amount that exist only in his chain store. The notification can be set to automatically display whenever that is a shortage or the administrator and supervisor can decide to click the notification at anytime they wish to e.g. at opening or closing time.

4.1.8 Help

This module contains the user manual that includes troubleshooting and instructions on how to use the system. This would help reduce any questions or doubts on how the system works.

4.2 Non – Functional Requirements

Non – functional requirements represent the restrictions and constraints on the system. They provide a boundary on the system by limiting the choices of operating systems, hardware and software tools which can be used by the system. Below are the non – functional requirements of the system.

4.2.1 Reliability

Reliability is the extent to which a system can be expected to perform its functions with accuracy and precision. Thus the system should be reliable in performing its routine functions and operations. For example, when a button is clicked in the Inventory Management System, it should be able to perform an action or generate a message to inform the user what is happening.

4.2.2 Usability

The system should be developed in a way that it is easy to use. It is meant to make the user comfortable when using the system and not encounter any difficulties. The Inventory Management System interface would make it easy to understand the functionalities of the system.

4.2.3 Security

The system should be equipped with sufficient security in par with the user access level. Each access by the user needs to be authenticated and validated by the system,

never showing any potential leakage of the system. On way of doing this is by encrypting the password.

4.2.4 Manageability

The modules within the system should be easy to manage, thus making the maintenance and enhancement works easier and not time consuming.

4.2.5 Flexibility

The system should have the capability of taking advantage of newer technologies and resources. This means that the Inventory Management System should be able to be implemented in a changing environment.

4.2.6 On Time

The system should be developed with in the given time frame. During this period, the requirements and testing should be completed.

4.2.7 Easy to Navigate

The navigation of the system should be made as simple as possible as it deals with not only with people who have adequate computer knowledge but also those with minimal computer literacy. Therefore buttons and icons must contain graphics or symbols that are easy to comprehend.

4.3 Chosen Development Tools

4.3.1 Operating System

MS Windows 2000 was chosen as the operating system for the development of the application. This is because comparatively with Linux and UNIX, MS Windows 2000 is user –friendly and provides easier navigation in the user interface. It is highly reliable as the source codes are difficult to obtain. It is also a cost-effective operating systems, which is within the budget of most companies.

4.3.2 Database Management System

The chosen DBMS is Microsoft SQL Server 2000. The reason this DBMS is chosen is that it has a higher database capacity and functions when compared to other DBMS like Microsoft Access and requires less configuration than Oracle 8i.

4.3.3 Authoring Tools

Visual Basic 6 will be used to develop the application. This is because Visual Basic is a programming tool which has powerful features like graphical user interfaces, event handling, object – oriented features, error handling and structured programming which proves suitable when developing the Inventory Management System.

4.3.4 Other Programming Tools

Crystal Reports, a report generating and management tool will be used to produce reports. It allows the creation of complete, customized, attractive management reports

quickly and easily. It has the capacity to process multiple and simultaneous jobs without compromising the performance of the report when opening or formatting it.

4.4 Hardware Requirements

- a) Type and Speed of Processor – 336 MHz and above
- b) Memory – 64 MB RAM and above
- c) Size of hard disk – 2GB with 300MB free disk space for Lotus Notes and Domino Designer program and data files.
- d) Operating System – MS Windows 2000 Professional
- e) Resolution of the screen – 800 x 600 pixels
- f) Number of colors on the screen – 256 and above

4.5 Information Gathering

4.5.1 Internet

The Internet is a very effective place to gather information concerning the system. There are many similar systems available on the net that proved to be useful and valuable to the development of this system. Some websites provided useful examples of inventory management that were used as a guideline to determine the functional requirements. Other websites offered vast information on the development tools needed to design and implement the system.

4.5.2 Books and Journals

Research was done by reviewing books and journals that contained relevant information needed for the system. The library provided a vast array of books that proved valuable especially when conducting the literature review.

4.5.3 Informal Interviews

Informal interviews and discussions were conducted with professionals in the related areas of Inventory Management. For example interviews with the staff of MPH Bookstore and 7-11 provided an important insight into the workings of an inventory management system in the real world.

4.5.4 Discussions with Lecturer

Discussions with the lecturer proved to be invaluable especially in helping to answer queries and clear doubts on the development of the system, especially in determining the requirements of the system.

4.6 Conclusion

This chapter is on determining the functional and non –functional requirements of the system. It also indicates the chosen development tools researched in Literature Review and the hardware requirements needed for the system. It includes important and valuable information gathered by interviews and discussions.

Chapter 5 – Systems Design

5.1 Process Modelling

Process modelling involves representing the functions and processes of a system graphically (Kendall & Kendall, 1999).

A process would receive data, transforms that data into an output using tools like a Data Flow Diagram (DFD). Therefore DFD in IMS describes the points where the information enters the process and the points where it comes out as an output.

Figure 5.1 shows the common notations found in a DFD diagram.

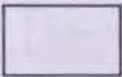


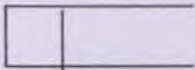
Symbols	Name	Description
	Entity	An entity can send data to the system or receive data from it. It can be a person, outside organisation or a different Information System. It is considered as outside of the boundaries of the system.
	Process	A process accepts input data and transforms the data into output data. It usually consists of a verb followed by a noun.
	Data Flow	Shows the movement of data from a source to a destination. The head of the arrow points to the destination.
	Data Store	Acts as a data repository that allows addition and retrieval of data.

Figure 5.1 Common Notations in Data Flow Diagrams

5.2 Data Flow Diagrams

5.2.1 Context Diagram

Context Diagrams are the highest levels of Data Flow Diagrams that encompasses the scope and boundaries of the system. For the Inventory Management System (IMS), the context diagram in figure 5.2 shows the main entities of the system that is the main office and the chain store. It also includes the main functions these entities perform with regards to the system.

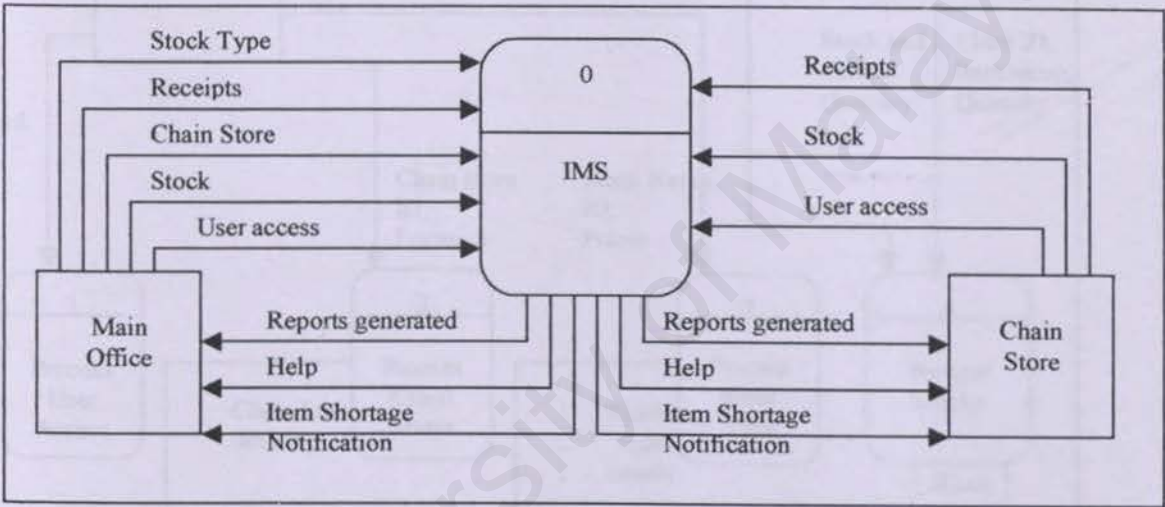


Figure 5.2 Context Diagram for Inventory Management System

5.2.2 Diagram 0

Figure 5.3 shows the Diagram 0 for IMS. Diagram 0 contains all the modules in the IMS, which are User Access Authentication, Chain Stores, Stocks, Stock Type, Receipts, Reports, Shortage Notification and Help.

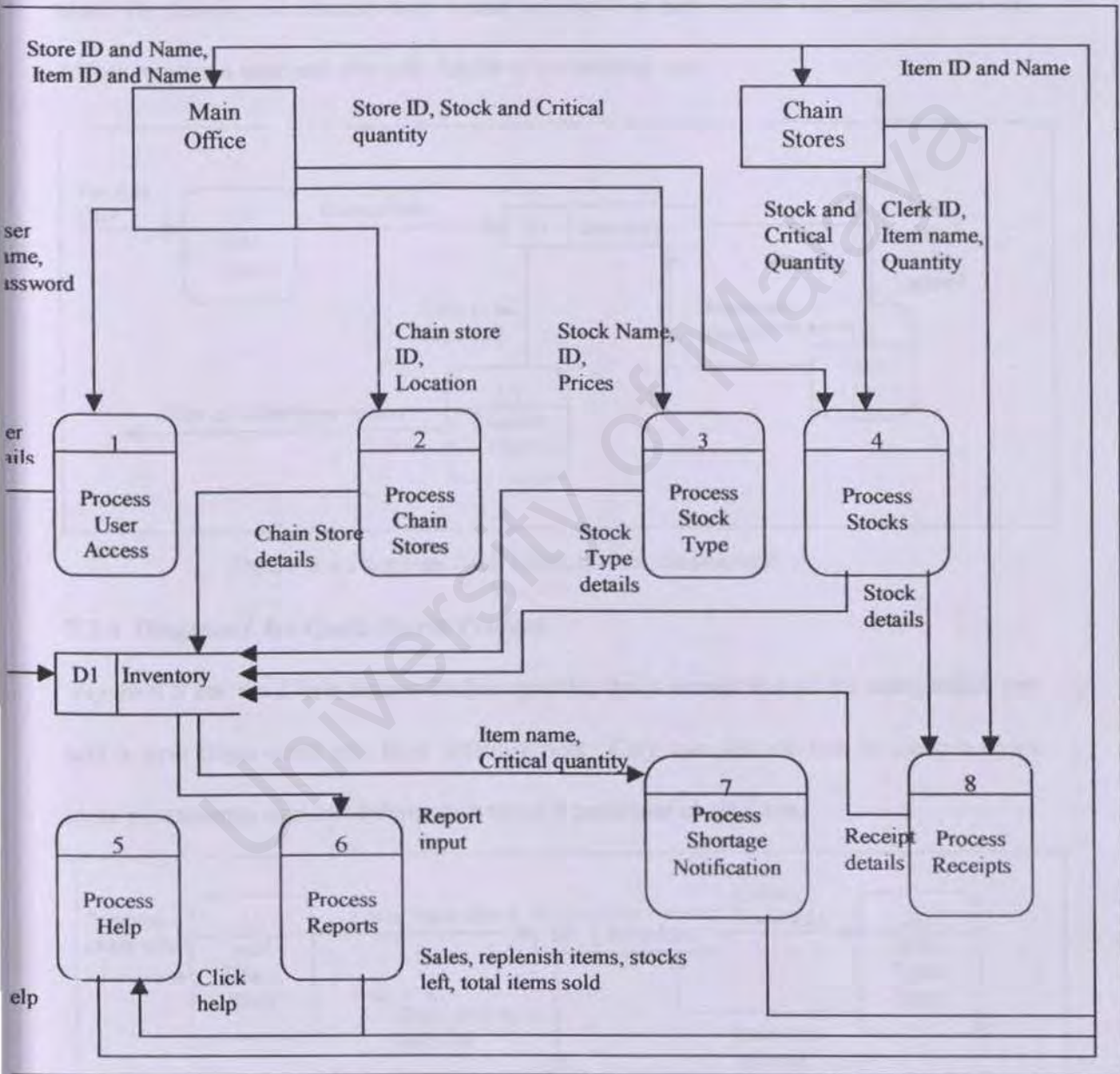


Figure 5.3 Diagram 0 for Inventory Management System

5.2.3 Diagram 1 for User Access Authentication

Figure 5.4 for the User Access Authentication, the administration at the main office adds a new user for the system based on the given roles of administrator, supervisor or clerk. They are given a login name and password that would indicate their respective chain store. By default, the administrator would be placed in main office. The administrator can add or remove a user and also edit details of an existing user.

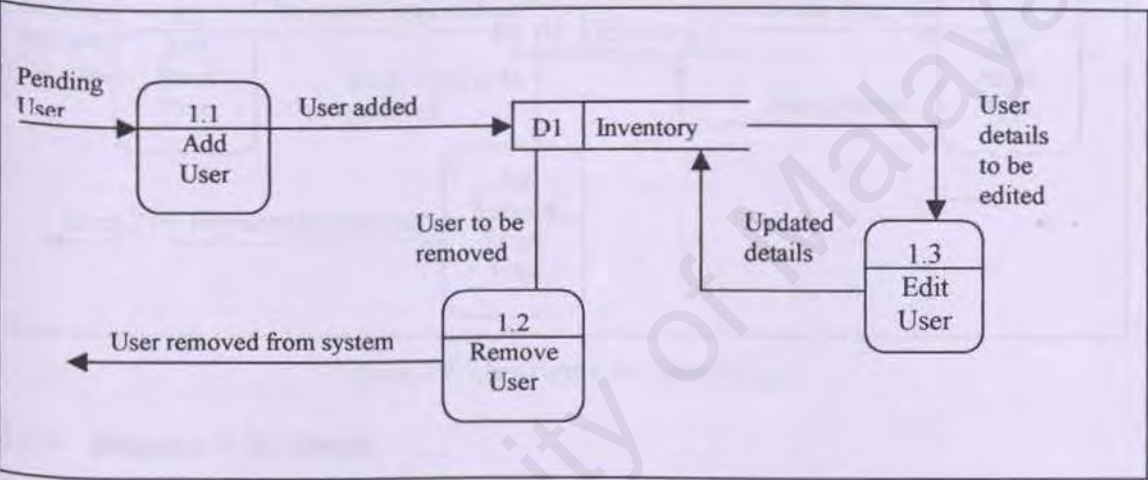


Figure 5.4 Diagram 1 for User Access Authentication

5.2.4 Diagram 1 for Chain Stores Profiles

Figure 5.5 for the Chain Stores Profile module, the administrator at the main office can add a new chain store into their establishment. They can also choose to close a chain store permanently and edit information about a particular chain store.

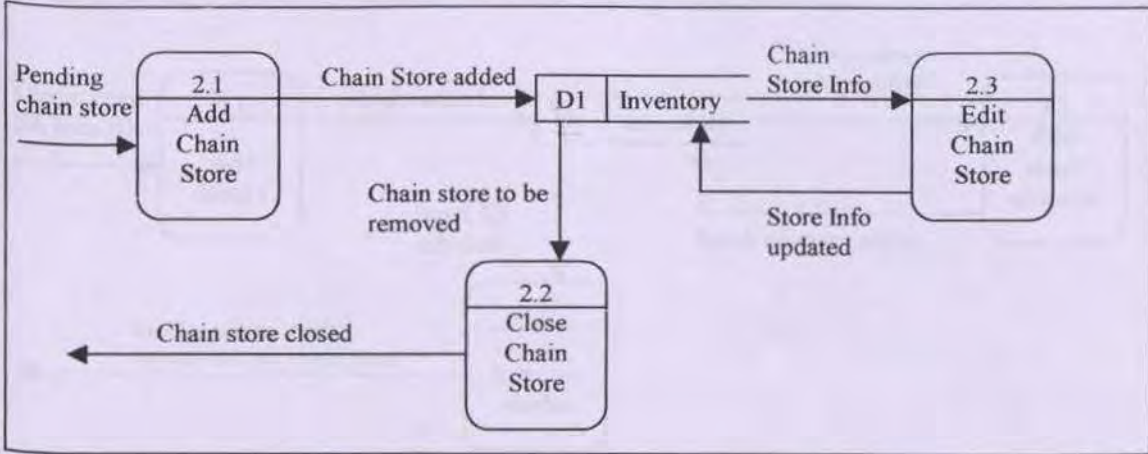


Figure 5.5 Diagram 1 for Chain Store Profile

5.2.5 Diagram 1 for Stock Type

Figure 5.6 for the Stock Type module, the administrator at the main office can add a new item into their establishment like milk powder or sweets. They can permanently remove an item from the company and edit details of the existing item like changing it weight or sales price.

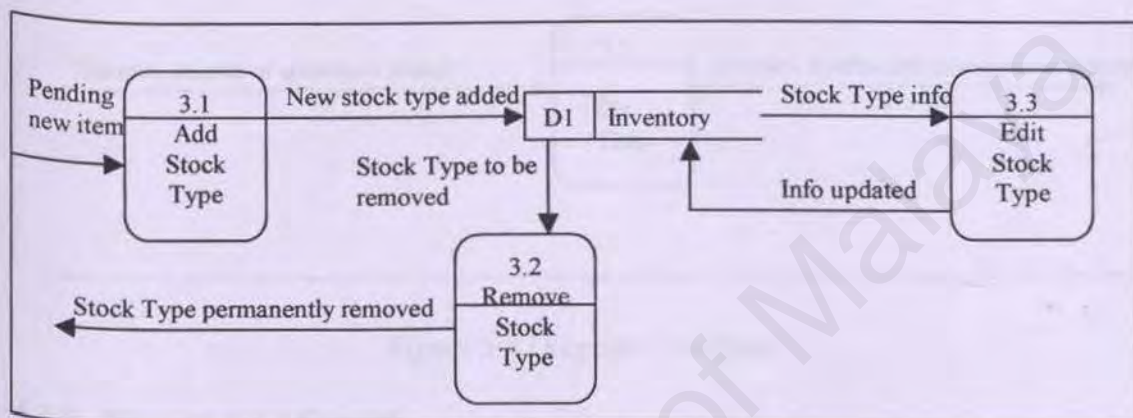


Figure 5.6 Diagram 1 for Stock Type

5.2.6 Diagram 1 for Stocks

Figure 5.7 for Stocks module, the administrator will now add the stocks of the item into the chain stores. They can remove the stocks as well. In the module, they and the supervisor decided the quantity of the stock in the chain store and the critical quantity. However the supervisor can only add the quantity of the existing stock only in his chain store. He can't add the stocks of a new item or remove it.

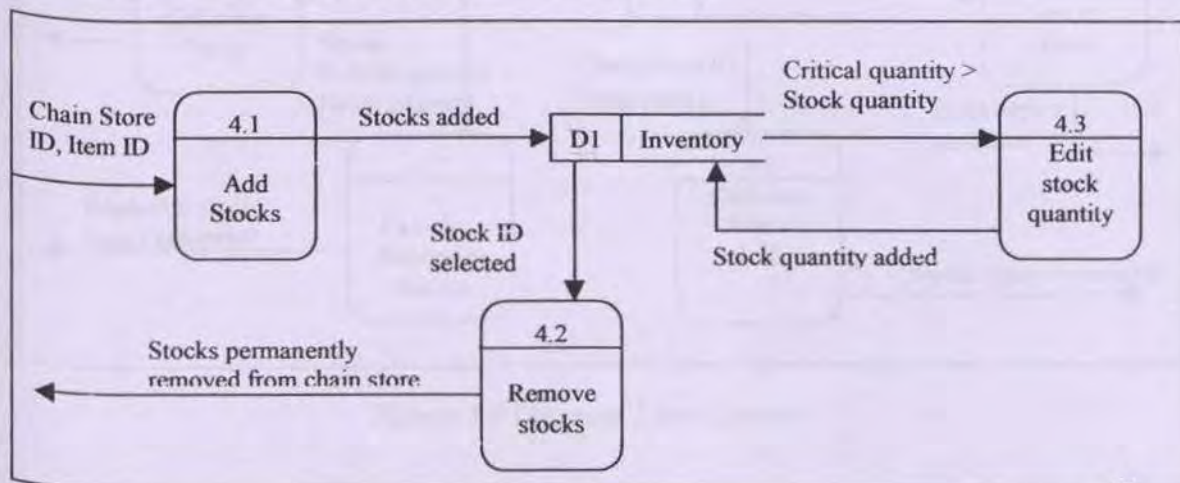


Figure 5.7 Diagram 1 for Stocks

5.2.7 Diagram 1 for Help

Figure 5.6 for Help module, both the administrator at the main office and the supervisors and clerks at the chain stores can click on the Help menu. This is to help them to seek answers on the functionality of the system.

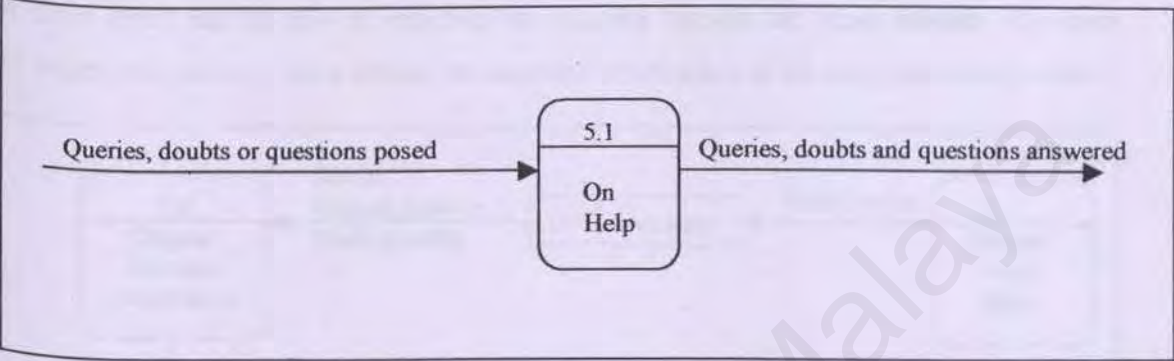


Figure 5.8 Diagram 1 on Help

5.2.8 Diagram 1 for Reports

Figure 5.7 for Reports module, both the administrator at the main office and supervisors at the chain stores can access this module. Among the reports generated are the sales of the chain stores, number of stocks left in the chain store, total stocks that need to be replenished and the number of items sold. The administrator can access the reports for all the chain stores but the supervisors can only access the reports of his respective chain store.

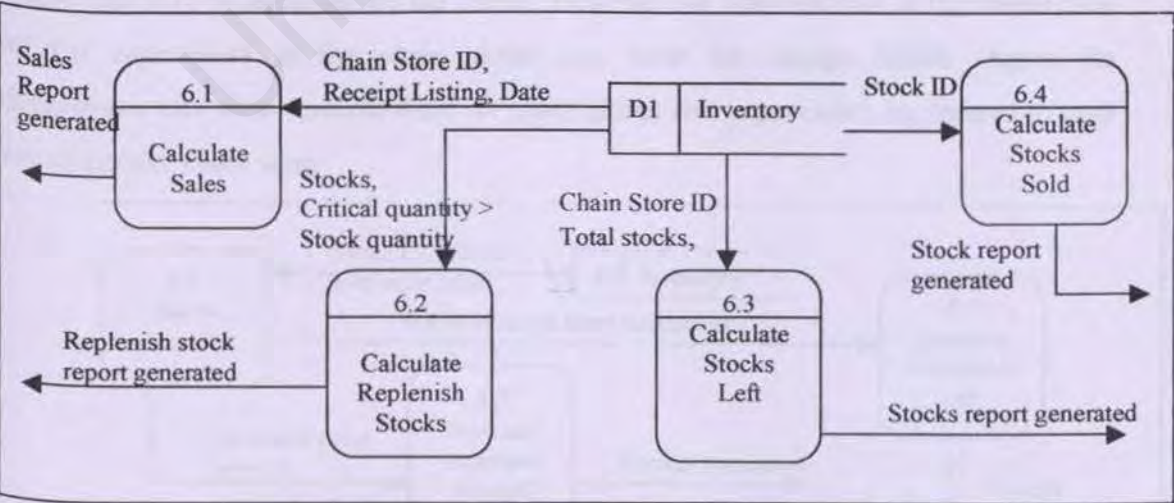


Figure 5.9 Diagram 1 for Reports

5.2.9 Diagram 1 for Shortage Notification

Figure 5.8 for Shortage Notification module, both the administrator at the main office and the supervisor at the chain store will be notified automatically or when the shortage notification icon is clicked, if the stocks of an item is below the critical quantity and inform them to replenish it. The administrator will have access to shortage notification on all chain stores and be able to replenish the quantity through the Stock module. However the supervisor can only have access the shortage notification of his respective chain store.

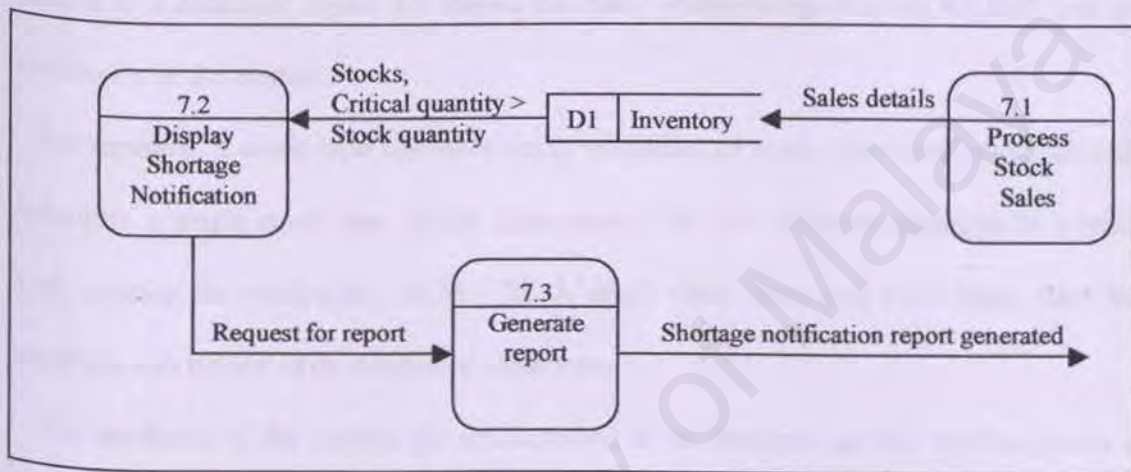


Figure 5.10 Diagram 1 for Shortage Notification

5.2.10 Diagram 1 for Receipts

Figure 5.9 for Receipts module, can be likened to a sales receipt at a check – out counter. A receipt is generated based on the sales of the stocks in the chain stores. This part of the module can only be accessed by the clerks. However the administrator at the main office and the supervisors at the chain stores can view the receipt details. Again the administrator can view receipts from all chain stores and supervisors are limited to only their respective chain store.

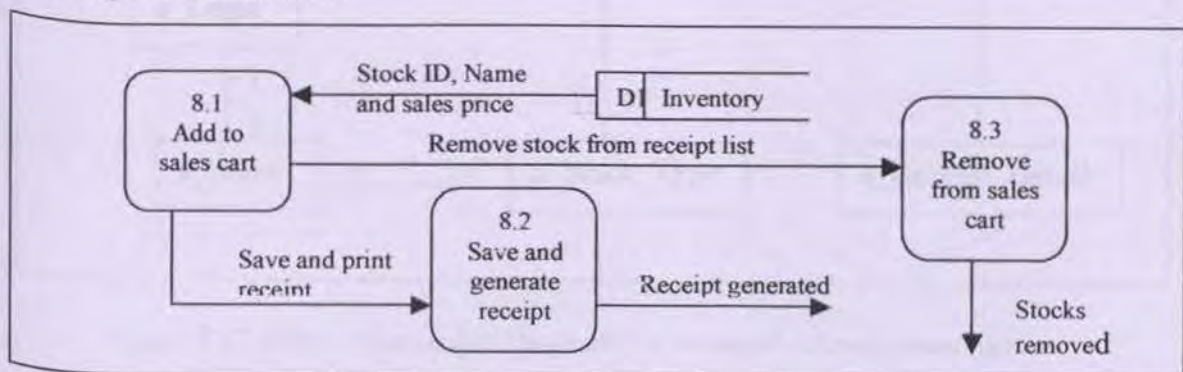


Figure 5.11 Diagram 1 for Receipts

5.3 Database Design

A database is used in the Inventory Management System (IMS). It is used to keep information of the chain stores, items, stock and sales of the chain stores.

5.3.1 Entity – Relationship Diagram

An entity – relationship diagram purpose is to show the relationship between the entities in a database. Figure 5.9 shows the entity –relationship diagram for IMS and the cardinality of the entities.

For example, a stock type can have many quantities of stock. However stock can only belong to a single stock type. Many chain stores can sell different quantities of a stock type, making the relationship an M – M. A single chain store may have many sales but each sale can belong to its respective chain store.

The attributes of the entities are not included in the database as they will be shown in the data dictionary below. This is also to avoid overcrowding the ER diagram.

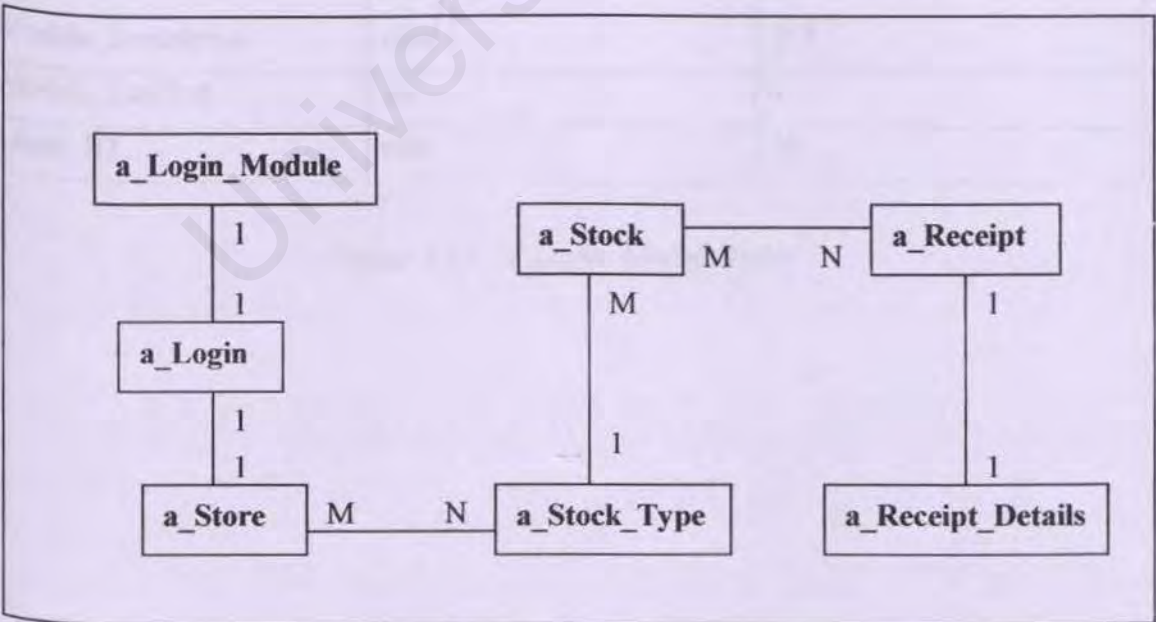


Figure 5.12 Entity Relationship Diagram for Inventory Management System

5.3.2 Data Dictionary

A data dictionary contains the tables needed to model a database. Below are the tables within the database named Inventory as shown in the Data Flow Diagram.

Name of Database : Inventory

Name of Table : a_Login

Name	Data Type	Size
Login_ID	char	10
Login_Name	char	30
Login_Password	char	10
Login_Level	char	255
Store_ID	char	10

Figure 5.13 : a_Login Table

Name of Table : a_Login_Module

Name	Data Type	Size
Module_Description	char	255
Module_Enabled	int	4
Login_ID	char	10

Figure 5.14 : a_Login_Module Table

Name of Table : a_Receipt

Name	Data Type	Size
Store_ID	char	10
Receipt_No	char	10
Receipt_Date	datetime	8
Total_Amount	money	8
Clerk_ID	char	10
Clerk_Name	char	255

Figure 5.15 : a_Receipt Table

Name of Table : a_Receipt_Details

Name	Data Type	Size
Store_ID	char	10
Receipt_No	char	10
Item_ID	char	10
Description	char	255
Quantity	numeric	9(18,0)
Unit_Price	money	8
Total_Amount	Money	8

Figure 5.16 : a_Receipt_Details Table

Name of Table : a_Stock

Name	Date Type	Size
Stock_ID	char	10
Stock_Description	char	255
Stock_Quantity	numeric	9(18,0)
Critical_Quantity	numeric	9(18,0)
Store_ID	char	10

Figure 5.17 : a_Stock Table

Name of Table : a_Stock_Type

Name	Data Type	Size
Stock_ID	char	10
Stock_Description	char	255
Stock_Weight	numeric	9(18,0)
Unit_of_Measurement	char	255
Cost_Price	money	8
Sales_Price	money	8

Figure 5.18 : a_Stock_Type table

Name of Table : a_Store

Name	Data Type	Size
Store_ID	char	10
Store_Name	char	255
Address	char	255
Telephone	char	255
Date_Opened	datetime	8
Supervisor_ID	char	10
Supervisor_Name	char	10
Shortage_Notification	int	4
Main_Office	int	4

Figure 5.19 : a_Store table

5.4 Interface Design

The interface of system can be considered as the representation of the system in the end user's eye. Therefore, the design of the interface has to be informative and attractive to the user, allowing him / her to navigate with ease when using the system. Graphics and

animation will be included with reasonable amount without compromising the response time or overcrowding the interface (Weinschenk & Jamar & Yeo, 1997).

Below are a few layouts of the interface designs for IMS. It is not a permanent design and is subject to change.

The 'Stock Type' window has a title bar with a dollar sign icon and a close button. It contains a 'Stock ID/Name' field with the value '0010'. Below this are two buttons: a checkmark and an 'X'. The main area is a table with two columns: 'Stock ID' and 'Stock Description'.

Stock ID	Stock Description
0001	Ais Kacang
0002	MILO
0005	CUP
0006	PLATE
0007	RULER
0008	BURGER
0011	SWEETS
0012	APPLES
0015	GUAVA
0016	MILK
0017	ORANGES

Figure 5.20 Interface Design for Stock Type

The 'Stock' window has a title bar with a dollar sign icon and a close button. It contains several input fields: 'Store ID/Name' (with a dropdown arrow), 'Supervisor ID/Name', 'Date Opened', and 'Telephone'. Below these are 'Stock ID/Name' (with a dropdown arrow), 'Stock Quantity', 'Weight', 'Cost Price', 'Date Manufactured' (with a date picker showing '28/01/2003'), 'Critical Quantity', 'Unit', 'Sales Price', and 'Use by Date' (with a date picker showing '29/01/2003'). There are also checkmark and 'X' buttons. The main area is a table with seven columns: 'Stock ID', 'Stock Name', 'Quantity', 'Critical Quantity', 'Weight', 'Unit', and 'Cost'.

Stock ID	Stock Name	Quantity	Critical Quantity	Weight	Unit	Cost
----------	------------	----------	-------------------	--------	------	------

Figure 5.21 Interface Design for Stocks

5.5 Conclusion

System design includes the design and planning of the modules within the system by implementing the Data Flow Diagrams. It also includes the design of the tables within the database and several interface designs of the module.

CHAPTER 6
SYSTEM
IMPLEMENTATION

Chapter 6 – System Implementation

6.1 Enhancements and Changes to the System

6.1.1 Enhancements on the Modules

A few notable changes were made to proposed system earlier on. This was to further enhance and improve the functionality of the Inventory Management System.

Modules that were added to the system include the User Authentication module that would authenticate whether or not the person logging on to the system is a valid user of the system.

The module added to the system is the Notification on Item Shortage module. This module would inform the supervisor and administrator if the quantity of an item were below the critical level. It can be set to automatically detect when an item is below the critical level or the supervisor and administrator can choose to view the notification at a specific time such as at closing time.

The other module is the Help module that would provide assistance in using the system. Other modules were renamed to easier identify them. The Item module was changed to Stock Type module. The Sales module was switched to Receipt module. This would help the users to recognize the functions with less difficulty.

6.1.2 Changes on the Programming Language

The previous proposed language of development was Lotus Notes R5 and Domino Designer. However the programming language was changed to Visual Basic 6. This is because Visual Basic has a vast range of online references when compared to Lotus Notes. It is easier to design the User Interface using Visual Basic therefore making it a suitable choice to develop the Inventory Management System.

6.1.3 Main purpose of Reports Generation

Being one of the most important modules on the IMS if not the most important, reports generated in the IMS have several important results as listed below.

The reports generated are a standard business format for gathering and structuring information with the means of relating information to the decision makers.

Reports also are a tool for making business decisions such as deciding on certain stock type and then distributing the quantities of the stock to the chain stores based on reports from the previous sales.

The reports generated could be considered as a tool for minimizing poor business decisions and help to justify certain business risks.

6.2 Development Environment

The development environment for the IMS consists of both the hardware and software configuration. Using the appropriate hardware and software is an important factor in determining the successfulness of a system.

6.2.1 Hardware Configuration

Intel Pentium 3 500 MHz processor

256 MB SD RAM

15 GB Hard disk

15" 256 – Colour monitor (800 x 600) resolution

1.44 MB Floppy drive

52 x CD- ROM Drive

Other standard peripherals

6.2.2 Software Configuration

The software tools used for the system development are vital to the successful implementation of the IMS. The software specifications used in the development of the system are listed below.

Software	Usage	Description
Microsoft Windows 2000 P	System Requirements	Operating System
Microsoft Visual Basic 6.0	System Requirements	Application Development
Microsoft SQL Server 2000	System Requirements	Database Server
Crystal Reports 8.5	System Requirements	Report Editor
Adobe Photoshop	System Requirements	Image Editor

Figure 6.1 Software Configuration for IMS

6.3 Coding of Modules in IMS

6.3.1 User Access Authentication

The valid user enters the login name and password to access the IMS and then clicks ok to enter the system. The login module verifies the user login and password against the database table a_Login to confirm the role of the user and the table a_Login_Module to determine the modules that that particular user can access. This is to increase the security level in IMS by making sure that only legitimate users can log on to the system

The SQL statement that opens the table a_Login to verify the login name and password
Coding principle in VB suggests that text fields start with 'txt' to standardize text field.

```
OpenSql "Select * From a_Login Where Login_Name=" + Trim(txtLoginName) + " and
Login_Password=" + Trim(txtPassword) + """
```

This code confirms the login id, login name, login level i.e. administrator, supervisor or clerk, the password and store id.

```
If Not sql.EOF Then
```

```
    Login_Info.LoginID = Trim(sql![Login_ID])
```

```
    Login_Info.LoginName = Trim(sql![Login_Name])
```

```
    Login_Info.LoginLevel = Trim(sql![Login_Level])
```

```
    Login_Info.LoginPassword = Trim(sql![Login_Password])
```

```
    Login_Info.Store_ID = Trim(sql![Store_ID])
```

```
Found = True
```


The SQL statement that opens the Login ID from table a_Login_Module

If Found Then

```
OpenSql "Select * From a_Login_Module Where Login_ID=" +  
Trim(Login_Info.LoginID) + ""
```

If Not sql.EOF Then

This block of code determines which module i.e. Reports, Receipt or Store Information is accessible based on the user role. The modules are coded in a format that is easy to read and execute

Do While Not sql.EOF

If Trim(sql![Module_Description]) = "Receipt" Then

Login_Info.urReceipt = sql![Module_Enabled]

ElseIf Trim(sql![Module_Description]) = "Report" Then

Login_Info.urReport = sql![Module_Enabled]

ElseIf Trim(sql![Module_Description]) = "Stock" Then

Login_Info.urStock = sql![Module_Enabled]

ElseIf Trim(sql![Module_Description]) = "Store Information" Then

Login_Info.urStoreInformation = sql![Module_Enabled]

ElseIf Trim(sql![Module_Description]) = "User Account" Then

Login_Info.urUserAccount = sql![Module_Enabled]

End If

sql.MoveNext

Loop

End If

CloseSql

6.3.2 Chain Store Profile

Only the administrator at the main office has access to this module whereby he can add, remove or edit the details of a chain store. For example, the code below shows how the administrator has to enter the details at the compulsory (*) fields as stated in the module. The information is then saved into the table a_Store in the database.

Upon clicking the Store Information:

If the administrator wishes to add a new chain store, he has to fill in compulsory fields like the store name and the address.

Administrator will be prompted to enter information into the compulsory fields.

Coding Principles in VB suggest that commands start with 'cmd' to coordinate the style of coding.

```
Private Sub cmdAdd_Click()
```

```
    If Trim(txtStoreID) = "" Then
```

```
        Focus txtStoreID
```

```
        Exit Sub
```

```
    End If
```

Input text fields are written in a similar pattern to standardize the pattern of coding

```
    If Trim(txtStoreName) = "" Then
```

```
        Prompt "Please enter store name !!", a_Warning
```

```
        Focus txtStoreName
```

```
        Exit Sub
```

```
    End If
```

```
    If Trim(txtAddress) = "" Then
```

```
        Prompt "Please enter the store address!!", a_Warning
```

```
        Focus txtAddress
```

```
        Exit Sub
```

```
    End If
```


The Store ID and Name, address, telephone number, date opened and supervisor id are listed into a listview function labeled L1. This makes the coding structure easier to understand and implement. The format for the date is entered as shown in Subitem 4 and the supervisor ID in SubItem 5 starts with the digit 0 as in 0001.

```
L1.ListItems.Add 1, , ""  
L1.ListItems(1).Text = Trim(txtStoreID)  
L1.ListItems(1).SubItems(1) = Trim(txtStoreName)  
L1.ListItems(1).SubItems(2) = Trim(txtAddress)  
L1.ListItems(1).SubItems(3) = Trim(txtTelephone)  
L1.ListItems(1).SubItems(4) = Format(txtDateOpened.Value, "dd/mm/yyyy")  
L1.ListItems(1).SubItems(5) = Format(txtSupervisorID, "0###")
```

The information above is then stored in the table by using RecordSet 'rst' by utilizing the ActiveX Data Objects in VB (ADO) . The function adds the information into the table as shown below. The code is implemented in the same order as in the listview to maintain synchronicity. The RecordSet is then updated and closed.

```
OpenRst "a_Store"  
rst.AddNew  
rst![Store_ID] = Trim(txtStoreID)  
rst![Store_Name] = Trim(txtStoreName)  
rst![Address] = Trim(txtAddress)  
rst![Telephone] = Trim(txtTelephone)  
rst![Date_Opened] = Format(txtDateOpened.Value, "dd/mm/yyyy")  
rst![Supervisor_ID] = Format(txtSupervisorID, "0###")  
rst.Update  
CloseRst
```

6.3.3 Stock Type

This module is accessible only by the administrator. He can add a new stock type such as Milo or Nestle milk, remove a stock type permanently from the company or edit the details of a current item.

When adding a new stock type, the administrator must fill in the compulsory fields such as Stock Name and Weight below otherwise he will be prompted to do so if the fields are left empty. These fields are set to compulsory and a Warning is issued to remind the administrator.

```
If Trim(txtStockName) = "" Then
    Prompt "Please enter stock description !!", a_Warning
    Focus txtStockName
    Exit Sub
End If
If Trim(txtWeight) = "" Then
    Prompt "Please enter the stock weight !!", a_Warning
    Focus txtWeight
    Exit Sub
End If
```

The Stock ID and Name, weight, unit, cost price and sales price are listed into a listview function labeled L1. The label L1 is still kept for the purpose of code maintenance. This makes the coding structure easier to understand and implement. The format for the price is shown in SubItem 4.

```
L1.ListItems.Add 1, , ""
L1.ListItems(1).Text = Trim(txtStockID)
L1.ListItems(1).SubItems(1) = Trim(txtStockName)
L1.ListItems(1).SubItems(2) = Val(txtWeight)
```



```

L1.ListItems(1).SubItems(3) = Trim(txtUnit)
L1.ListItems(1).SubItems(4) = Format(txtCostPrice2, "0.##0")
L1.ListItems(1).SubItems(5) = Format(txtSalesPrice, "0.##0")

```

The information above is then stored in the table a_Stock_Type by using RecordSet rst by utilizing the ActiveX Data Objects in VB (ADO) . The same function 'rst' is coded in a similar style to the code written in Chain Store Profiles.

```

OpenRst "a_Stock_Type"
rst.AddNew
rst![Stock_ID] = Trim(txtStockID)
rst![Stock_Description] = Trim(txtStockName)
rst![Stock_Weight] = Trim(txtWeight)
rst![Unit_Of_Measurement] = Trim(txtUnit)
rst![Cost_Price] = Trim(txtCostPrice2)
rst![Sales_Price] = Trim(txtSalesPrice)
rst.Update
CloseRst

```

6.3.4 Stocks

For this module, the administrator can add stocks of a stock type to the chain store. The administrator can add new stocks or remove them. However the supervisor of the chain store can add the current stock quantity once it is below the critical level for only his respective chain store. The administrator has access to add the stock quantity for all chain stores.

When the Stocks module is clicked, the form loads based on the role of the user. If the user has the role administrator, then he can select which chain store to add the stocks. However this particular sub module is disabled if the user role is supervisor whereby the

supervisor's chain store information appears by default, disabling him from adding stocks for other chain stores.

The SQL statement if the role is Administrator.

If Login_Info.mainOff And Trim(Login_Info.LoginLevel) = "Administrator" Then

OpenSql "Select a_Store.Store_ID From a_Store Order By [Store_ID]"

Else

If the role is Supervisor, then the following fields are disabled. The code is implemented in a similar pattern and written in a standard format.

txtStockID.Enabled = False

txtStockQuantity.Enabled = False

txtReorderQuantity.Enabled = False

txtWeight.Enabled = False

txtUnit.Enabled = False

txtCostPrice.Enabled = False

txtSalesPrice.Enabled = False

The SQL statement if role is Supervisor. The Login Info is added to determine if the role is Supervisor

OpenSql "Select a_Store.Store_ID From a_Store Where Store_ID=" +

Trim(Login_Info.Store_ID) + " Order By [Store_ID]"

The supervisor cannot delete a stock type from his chain store and cannot add a new stock type to his chain store. Different functions like commands and texts are grouped by a block of code as below to maintain a good flow of code.

If Trim(Login_Info.LoginLevel) = "Supervisor" Then

cmdDelete.Enabled = False

txtStockID.Enabled = False

Exit Sub

End If

This code enables the supervisor and administrator at the main office to edit current quantity and critical quantity based on the Login Info

```
If (Trim(Login_Info.LoginLevel) = "Supervisor") Or (Login_Info.mainOff And  
Trim(Login_Info.LoginLevel) = "Supervisor") Or (Login_Info.mainOff And  
Trim(Login_Info.LoginLevel) = "Administrator") Then  
txtStockQuantity.Enabled = True  
txtReorderQuantity.Enabled = True
```

```
End If
```

Both the supervisor and administrator cannot edit the weight, unit, cost price and sales price field because this information is static to a certain stock type.

```
If Login_Info.mainOff Then  
txtWeight.Enabled = False  
txtUnit.Enabled = False  
txtCostPrice.Enabled = False  
txtSalesPrice.Enabled = False
```

```
End If
```

The SQL statement to open table a_store to retrieve data, store id and name, supervisor name and id, date store opened, telephone number. The SQL statement sets the information by Store ID and returns the Store Name, Supervisor ID and Name as shown below.

```
OpenSql "Select * From a_Store Where Store_ID='" + Trim(txtStoreID.Text) + ""  
Order By [Store_ID]"
```

```
If Not sql.EOF Then
```

```
txtStoreName = Trim(sql![Store_Name])  
txtSupervisorID = Trim(sql![Supervisor_ID])  
txtSupervisorName = Trim(sql![Supervisor_Name])
```

```
End If
```

```
CloseSql
```

This SQL statement is for two tables; a_Stock and a_Stock_Type. The SQL retrieves the data from both the tables based on their common attributes that is Stock ID and Stock Description. This is to list various information into one listview L1. Similar SQL format is maintained although query is from 2 tables.

```
OpenSql                                "Select                                a_Stock.*,
a_Store.Store_Name,a_Stock_Type.Stock_Weight,a_Stock_Type.Unit_of_Measurement,
a_Stock_Type.Cost_Price,a_Stock_Type.Sales_Price From a_Stock_Type,a_Stock,
a_Store Where a_Stock.Store_ID=a_Store.Store_ID AND a_Stock.Stock_ID =
a_Stock_Type.Stock_ID and a_Stock.Store_ID=" + Trim(txtStoreID.Text)
+ " Order by a_Stock.[Stock_ID], a_Stock.[Store_ID]"
```

6.3.5 Receipts

This module can be likened to the sales receipt when an item is purchased at the chain store. The clerks at the chain store will click on a certain stock type, enter the quantity purchased and then click to save and print the receipt. The supervisor and administrator cannot access this part of the module. They however can view all the receipts generated by the sales of the stocks.

The clerk selects the stock type by clicking on the Stock ID / Description. This opens the stock listing module that displays all the stocks available in the chain store.

With frmReceipt

```
.txtStockIDDescription = Trim(L1.SelectedItem.Text) + "/" +
Trim(L1.SelectedItem.SubItems(1))
```

If the clerk want to select a stock type, that particular stock type must not have been selected before in the same receipt which means that UsedQuantity = 0. The stock type is then added to the listview L1.

Dim UsedQuantity As Long

```
UsedQuantity = 0
```

```
For I = 1 To .L1.ListItems.Count
```

```
If Trim(.L1.ListItems(I).Text) = Trim(L1.SelectedItem.Text) Then
```



```

        UsedQuantity = Val(.L1.ListItems(I).SubItems(2))
    Exit For
End If
Next

```

This particular code validates the current quantity of a stock after a certain quantity of that stock has been sold. It also displays the unit price and sub total.

```

.txtQuantity.Tag = Val(L1.SelectedItem.SubItems(2)) - UsedQuantity
.txtQuantity.ToolTipText = " Quantity on hand : " +
Trim(Str(Val(L1.SelectedItem.SubItems(2)) - UsedQuantity)) + " !! "
.txtQuantity = ""
.txtUnitPrice = Format(L1.SelectedItem.SubItems(4), "0.#0")
.txtSubTotal = ""
End With

```

This code would issue a warning if quantity entered is more than the existing quantity. This is maintain the integrity of the functional requirement as there can be no negative values for the stock quantity.

```

If Val(txtQuantity) <= 0 Or Val(txtQuantity) > Val(txtQuantity.Tag) Then
    Prompt "Out-standing sales quantity, It must not greater than (" +
Trim(Str(txtQuantity.Tag)) + ") !!", a_Warning
    Focus txtQuantity
Exit Sub
End If

```

When receipt is clicked to be saved and printed, the SQL statement updates the latest info on stock quantity to table a_Stock. The format for Execute is parallel to OpenSQL

```

ExecuteSql "Update a_Stock Set Stock_Quantity=Stock_Quantity-" +
Trim(L1.ListItems(I).SubItems(2)) + " Where Stock_ID=" + Trim(L1.ListItems(I).Text)
+ " and Store_ID=" + Trim(txtStoreID) + ""

```

This code enables the Crystal Report function CR to access tables a_Receipt, a_Receipt_Details, a_Store for Crystal Reports. The function is labeled as CR so that it's easier to recognize as the initials of Crystal Reports and makes coding simpler.

```
CR.DataFiles(0) = "IMS.dbo.a_Receipt"
```

```
CR.DataFiles(1) = "IMS.dbo.a_Receipt_Details"
```

```
CR.DataFiles(2) = "IMS.dbo.a_Store"
```

This code establishes the connection to Crystal Reports and MS SQL server by requesting the Data Source Name (DSN), ServerName(PC Name), Login Name and Password.

```
CR.Connect = "DSN=" + ServerName + ";UID=" + ServerLoginName + ";PWD=" +  
ServerPassword + ";DSQ=" + DatabaseName
```

```
TempReturnValue = CR.LogOnServer(GetWinDir + "\Crystal\P2ssql.dll", ServerName,  
DatabaseName, ServerLoginName, ServerPassword)
```

When the clerk clicks 'Print Receipt', his code opens Crystal report file Receipt.rpt and set the selection formula for the report.

```
CR.ReportFileName = App.Path + "\Report\Receipt.rpt"
```

```
CR.SelectionFormula = "{a_Receipt.Receipt_No}=" + TempReceiptID + ""
```

```
CR.Destination = crptToWindow
```

```
CR.Action = 1
```

This SQL statement determines that the administrator has access to all chain stores to view the receipts.

```
If Login_Info.LoginLevel = "Administrator" Then
```

```
OpenSql "Select * From a_Receipt Order by [Store_ID]"
```

This SQL statement states that the supervisors can only view the receipts for their respective chain store as determined by his Login Info.

```
OpenSql "Select * From a_Receipt Where Store_ID=" +
```

```
Trim(Login_Info.Store_ID) + " Order by [Store_ID]"
```


6.3.6 Shortage Notification

This module is meant to notify the supervisors and administrators that a stock type is below the critical level and must be replenished immediately. The supervisors would receive shortage notification of stocks related to their respective chain store only. The administrator will be able to view all the stocks that are below the critical level for all chain stores. The shortage notification is flexible in the sense that it can be set to automatically appear when the supervisor or administrator logs in, or they can select to view the notification at a certain time such as opening or closing time.

The SQL statement to notify on stock shortage determines the login level. Supervisors can view the stocks in their chain store and administrator can view all stocks below the critical level in all chain stores.

If Trim(Login_Info.LoginLevel) = "Administrator" Then

OpenSql "Select a_Stock.*, a_Store.Store_Name From a_Stock, a_Store Where

a_Stock.Stock_Quantity <= a_Stock.Critical_Quantity and

a_Stock.Store_ID=a_Store.Store_ID Order by a_Stock.[Stock_ID],

a_Stock.[Store_ID]"

Else

Supervisors role is again determined by their Login Info

OpenSql "Select a_Stock.*, a_Store.Store_Name From a_Stock, a_Store Where

a_Stock.Stock_Quantity <= a_Stock.Critical_Quantity and

a_Stock.Store_ID=a_Store.Store_ID and a_Stock.Store_ID="" +

Trim(Login_Info.Store_ID) + "" Order by a_Stock.[Stock_ID], a_Stock.[Store_ID]"

End If

6.3.7 Reports

This module is perhaps the most important module within the system. This is because the reports generated from the raw data of the transaction and sales will be transformed into vital information that is then utilized as knowledge to make business decisions. The supervisors have the flexibility of generating reports respective to their chain store and the administrators have access to reports generation for all the chain stores.

The reports generated are:

- i) The total sales of Chain Stores
- ii) Stocks left in the Chain Stores
- iii) Stocks below the critical quantity
- iv) Number of stock type sold in the Chain Stores.

i) The total sales of the Chain Stores

When the module loads, the administrator can have the option of selecting to view the sales of a particular chain store or the sales of all the chain stores. Coding Principles in VB suggest that combo box and checkbox be written with 'cbo' and 'chk' to maintain synchronicity. It also makes the functions easier to be identified by their controls.

```
If Login_Info.mainOff And Trim(Login_Info.LoginLevel) = "Administrator" Then
```

```
    cboStoreID.Clear
```

```
    OpenSql "Select * From a_Store Order By [Store_ID]"
```

```
    While Not sql.EOF
```

```
        cboStoreID.AddItem If(IsNull(sql![Store_ID]), "", Trim(sql![Store_ID]))
```

```
        sql.MoveNext
```

```
    Wend
```

```
    CloseSql
```

```
    cboStoreID.Enabled = True
```

```
    chkAll.Visible = True
```


This SQL statement executes when the Login Level is Supervisor for report generation

cboStoreID.AddItem Login_Info.Store_ID

OpenSql "Select * From a_Store Where [Store_ID]='" & Trim(Login_Info.Store_ID)
& """

The Store Name and Phone number appears upon selection of Store ID

txtName.Text = IIf(IsNull(sql![Store_Name]), "", Trim(sql![Store_Name]))

txtPhone.Text = IIf(IsNull(sql![Telephone]), "", Trim(sql![Telephone]))

This code allows the administrator to view the sales of all the chain stores

If chkAll.Visible Then

chkAll.Value = 1

The report formula for the sales of all chain stores. The format for the report formula is standardized whereby the tables and its attributes are set into the Crystal Report Window.

ReportFormula = "(Date({a_Receipt.Receipt_Date}) >= Date(" +

Format(txtFrom.Value, "yyyy,mm,dd") + ") and Date({a_Receipt.Receipt_Date}) <=

Date(" + Format(txtTo.Value, "yyyy,mm,dd") + ")) and

IsNull({a_Receipt.Supervisor_ID})"

If a Chain Store ID is selected to print a report on one chain store

If Trim(cboStoreID.Text) = "" Then

Prompt "Please select store id before print !!", a_Warning

cboStoreID.Enabled

End If

Report formula for sales of a single chain store based on the Store ID, based on the Store ID of the Receipt.

ReportFormula = "Trim({a_Receipt.Store_ID}) = '" & Trim(cboStoreID.Text) & "'"

AND (Date({a_Receipt.Receipt_Date}) >= Date(" +

Format(txtFrom.Value, "yyyy,mm,dd") + ") and

```
Date({a_Receipt.Receipt_Date}) <= Date(" + Format(txtTo.Value,
"yyyy,mm,dd") + ") and IsNull({a_Receipt.Supervisor_ID})"
```

Opens the report file TotalSalesOfEachChain.rpt in the Crystal Reports Window

```
CR.ReportFileName = App.Path + "\Report\TotalSalesOfEachChain.rpt"
```

```
CR.WindowTitle = " Total Sales of Each Chain Stores "
```

```
CR.ReplaceSelectionFormula ReportFormula
```

ii) Stocks Left in Chain Stores

When the module loads, the administrator can generate reports on all the stocks and quantities in a particular chain store or he can choose to view all the stocks in all chain stores.

```
If Login_Info.mainOff And Trim(Login_Info.LoginLevel) = "Administrator" Then
```

```
    cboStoreID.Clear
```

```
    OpenSql "Select * From a_Store Order By [Store_ID]"
```

```
    While Not sql.EOF
```

```
        cboStoreID.AddItem If(IsNull(sql![Store_ID]), "", Trim(sql![Store_ID]))
```

```
        sql.MoveNext
```

```
    Wend
```

```
    CloseSql
```

The supervisor can generate the same report, but he is limited to his respective chain store. The SQL statement determines that the Login Info is Supervisor, thus making the reports generated is based on his chain store by default.

```
OpenSql "Select * From a_Store Where [Store_ID]='" & Trim(Login_Info.Store_ID)
& """
```


If a single chain store is selected by its Store ID, the report is generated for that particular chain store. The report formula selects the Store Id from the table a_Stock.

```
If Trim(cboStoreID.Text) = "" Then
```

```
    cboStoreID.SetFocus
```

```
Exit Sub
```

```
End If
```

```
ReportFormula = "Trim({a_Stock.Store_ID})=" & Trim(cboStoreID.Text) & ""
```

The report formula for all chain stores is the same as the formula for one chain store. This is because the Login Level has been determined before the formula executes.

```
ReportFormula = "Trim({a_Stock.Store_ID})=" & Trim(cboStoreID.Text) & ""
```

```
End If
```

Open report file name StockLeft.rpt and Open in Crystal Reports Window

```
CR.ReportFileName = App.Path + "\Report\StockLeft.rpt"
```

```
CR.WindowTitle = " Total Stock Left in Each Chains Stores "
```

```
CR.ReplaceSelectionFormula ReportFormula
```

iii) Stocks that need to be replenished.

Upon clicking the module above, the shortage notification listing would appear as stated in 6.3.6. However upon clicking ok at the preview, the total stocks to replenished report would be generated.

The function CR opens the report file ShortageItems.rpt

```
CR.ReportFileName = App.Path + "\Report\ShortageItems.rpt"
```

If the Login Level is Administrator, then he can view the stocks that need to be replenished in all chain stores based on the selection formula. The notification is realized when the current quantity is less than the critical quantity.

If Trim(Login_Info.LoginLevel) = "Administrator" Then

CR.SelectionFormula = "{a_Stock.Stock_Quantity} <= {a_Stock.Critical_Quantity}"

If the Login Level is Supervisor, then he is able to view the stocks in his respective chain store only based on the Login Info.

CR.SelectionFormula = "{a_Stock.Stock_Quantity} <= {a_Stock.Critical_Quantity}

and {a_Stock.Store_ID}=" + Trim(Login_Info.Store_ID) + ""

iv) The number of stock types sold

This particular module would display the various stock types (items) that were sold in the chain stores from a certain date to another date. Reports can be generated for either one particular stock type or for all the stock types.

The code is implemented so that if one particular stock type is selected, then the stock name is displayed automatically.

If Trim(cboStockID.Text) <> "" Then

OpenSql "Select * From a_Stock_Type Where [Stock_ID]=" &

Trim(cboStockID.Text) & ""

Displays stock name automatically

If Not sql.EOF Then

txtStockName.Text = IIf(IsNull(sql![Stock_Description]), "",

Trim(sql![Stock_Description]))

Else

txtStockName.Text = ""

End If

CloseSql

End If

If all stock types are selected, then check all value to generate report for all stock types
The labels such as Stock ID and Name are disabled. Coding Principles in VB suggests
that labels be written with 'lb' to standardize coding methods.

```
If chkAll.Value = 1 Then
```

```
    cboStockID.Enabled = False
```

```
    cboStockID.ListIndex = -1
```

```
    txtStockName.Text = ""
```

```
    txtStockName.Enabled = False
```

```
    chkAll.Value = 1
```

```
    lbStockID.Enabled = False
```

```
    lbStockName.Enabled = False
```

The report formulas for one stock type and for all stock types are the same. This is
because access to the stock types was determined earlier based on the Login Info. The
table a_Receipt and its attribute Item_Id is set in the Crystal Report window.

```
ReportFormula = "Trim({a_Receipt_Details.Item_ID})=""" &
```

```
Trim(cboStockID.Text) & """"
```

Connect to Crystal Report, report on Number of Items sold

```
CR.ReportFileName = App.Path + "\Report\NoItemSold.rpt"
```

```
CR.WindowTitle = "Number of Stock Type in Chain Stores "
```

```
CR.ReplaceSelectionFormula ReportFormula
```

6.3.8 Help

This module is accessible by all user roles such as Administrators, Supervisors and
Clerks. This module contains the user manual that is embedded by using the OLE in
Visual Basic.

6.4 Conclusion

System Implementation states how the modules are implemented and executed based on the coding of the system. It also includes explanations on how the codes are written in the development tools to give a clear and complete understanding on how the IMS system functions.

CHAPTER 7

SYSTEM TESTING

Chapter 7 – System Testing

7.1 Software Testing on Inventory Management System (IMS)

Software Testing is an essential part of the system development. The goals of software testing is mainly on error detection which involves identifying errors within the system, error removal that is eradicate the mistakes that previously existed in the system and error tracking that is finding and correcting the errors itself (*Alka Jarvis and Vern Crandall, 1997*).

When testing the system, several testing principles are suggested as below

- i) Tests, be it unit tests or integration test should be planned properly and thoroughly before the actual testing begins.
- ii) All tests should be traceable to the user requirements itself. This simply means that the system must be validated against the user requirements.
- iii) Use of an independent 3rd party to test the system. This is to eliminate biasness and partiality because if the system was tested by those who actually developed the system, then they would know exactly how the system functions. Hence they would interact with the system in the same way as they developed it, making it harder to recognise errors.
- iv) Use the Pareto principle. The principle tells that 80 percent of all undetected errors are traceable to 20 percent of all modules. This means that by just isolating and testing 20 percent modules, 80 percent of the errors can be eliminated.

7.2 Unit Testing of Modules in IMS

Unit testing involves testing individual modules, or small clusters of modules. The purpose of unit testing is to thoroughly exercise individual classes to ensure high quality in their design and implementation. Unit testing is used to uncover defects that often do not appear in integration testing, as only a small subset of inputs and decision paths are used at this level. The three main types of unit testing are: ad hoc testing, white box testing and black box testing

7.2.1 Ad Hoc Testing on IMS

Ad Hoc testing is basically testing a system in order to find errors or mistakes within the system. Although this testing method was used in IMS, the results were not sufficient because is that although no mistakes or errors were found didn't mean that they do not exist. Ad Hoc testing did not manage to cover the entire functionality and coding within IMS.

Therefore, it proved better to support this method of testing with White Box and Black Box testing.

7.2.2 White Box Testing on IMS

White Box testing within the IMS system involved testing the structure of the codes inside the modules of the system. White Box testing involves several parts as tested below.

7.2.2.1 Segment Coverage

In segment coverage, each segment of code between the control structures was executed at least once to see if that function was actually performed. For the code below, the administrator or supervisor has to select a store id before adding the stocks for a particular stock type. Segment coverage involves testing that the particular code is carried out.

```
If Trim(txtStoreID) = "" Then
    Prompt "Please select store id before adding any stock !!", a_Warning
    txtStoreID.SetFocus
    Exit Sub
End If
```

7.2.2.2 Node Testing

In node testing, a particular part of a code in IMS is tested in every possible direction at least once such as what triggered that particular code and what did that code activate. In the Stock module, in order to add the stocks for that chain store, the administrator has to

first select a store id. The code to select the store id should then invoke the code to select stock id.

If Login_Info.mainOff And Trim(Login_Info.LoginLevel) = "Administrator" Then

OpenSql "Select a_Store.Store_ID From a_Store Order By [Store_ID]"

After selecting the store id, then only can the administrator select a stock. Node testing confirmed that stock id can't be selected without selecting the store id first.

OpenSql "Select a_Stock.*,

a_Store.Store_Name,a_Stock_Type.Stock_Weight,a_Stock_Type.Unit_of_Measurement,

+ Trim(txtStoreID.Text) + "" Order by a_Stock.[Stock_ID], a_Stock.[Store_ID]"

Upon selecting the stock ID, the code then triggeres another code to execute that is to list the stock details into a listview as shown below.

L1.ListItems.Add 1, , ""

L1.ListItems(1).Text = Trim(sql![Stock_ID])

L1.ListItems(1).SubItems(1) = Trim(sql![Stock_Description])

L1.ListItems(1).SubItems(2) = Trim(sql![Stock_Quantity])

7.2.2.3 Compound Condition Coverage

This particular test is used to examine multiple conditions whereby the conditions must be tested in one direction and then the opposite direction to see if it still proved to be true either way. Results for condition $a_Stock.Stock_Quantity \leq a_Stock.Critical_Quantity$ that would trigger the shortage notification is shown below.

	\leq Critical Qauntity	$>$ Critical Quantity
Stock Quantity	Notification = True	Nctification = False

Figure 7.1 Truth Table for Condition Coverage

7.2.2.4 Data Flow Testing

The main purpose of the data flow testing in IMS is to determine the flow of the codes within the module. Specific variables such as txtStoreName and txtAddress are tracked to reflect dependencies of the variables. This approach tends to uncover anomalies such as variables that are declared but not initialised. Therefore data flow testing proved important in determining that all variables that were declared in the modules in IMS are actually running when called or executed.

7.2.2.5 Basic Path Testing

This particular testing method tests the paths through which the codes defined and covered. However if there are dependencies within the code, then these dependencies must be tested as well. In IMS, upon logging in, the user is identified by their various roles such as administrator, supervisor or clerk. Therefore the users are directed into separate paths within the system based on their roles. Basic path testing was performed on the code below to verify that the users enter the same system although they have access to different modules within the system.

```
OpenSql "Select * From a_Login_Module Where Login_ID=" + Login_Info.LoginID)
Do While Not sql.EOF
    If Trim(sql![Module_Description]) = "Receipt" Then
        Login_Info.urReceipt = sql![Module_Enabled]
    ElseIf Trim(sql![Module_Description]) = "Report" Then
        Login_Info.urReport = sql![Module_Enabled]
    ElseIf Trim(sql![Module_Description]) = "Stock" Then
        Login_Info.urStock = sql![Module_Enabled]
    ElseIf Trim(sql![Module_Description]) = "Store Information" Then
        Login_Info.urStoreInformation = sql![Module_Enabled]
    ElseIf Trim(sql![Module_Description]) = "User Account" Then
        Login_Info.urUserAccount = sql![Module_Enabled]
    End If
```


7.2.2.6 Loop Testing

This testing strategy focuses on testing single loops such as WHILE loops and FOR loops and nested loops. Loop testing is performed on the codes within IMS to determine that loop actually run as stated within the code. For example, the FOR loop below searches to determined if a Supervisor ID already exists before an ID is assigned to a new supervisor.

```
For I = 1 To L1.ListItems.Count
```

```
  If Trim(L1.ListItems(I).Text) <> Trim(txtStoreID) Then
```

```
    If Trim(L1.ListItems(I).SubItems(5)) = Format(txtSupervisorID, "0###") Then
```

```
      Found = True
```

```
Exit For
```

7.2.3 Black Box Testing

Black box testing is basically testing the modules within the system without knowing the logical structure of the codes within the system, hence the term 'black box' whereby the users focus on testing the functional requirement, without knowing the internal syntax and codes of the system. . White Box testing involves several parts as tested below.

7.2.3.1 Error Guessing

This particular approach involves writing test cases that test the modules of the IMS. This method of testing, similar to Ad Hoc testing proved to be quite dependable in displaying errors in modules such as Stock whereby the supervisor cannot select another chain store to add stocks into it, he can only add stock to his respective chain store as stated in the functional requirement. This error has since been rectified.

7.2.3.2 Domain Testing

Domain testing is an important testing method for the IMS because it tests the dependencies of the modules based on a 'cause – and –effect' analysis. In the IMS, for Stock Module, when a new stock type is entered or if the quantity of an existing stock in the chain store is added, it affects the Receipt module whereby the sales and transaction

of the stocks occur. Therefore it is important to extend the test to other modules that are affected by data change of a particular module like the Stock Module to confirm data synchronicity.

7.2.3.3 Module Interface Testing

This method of black box testing attempts to test if whether the values in the interface are correct because they relate to the modules that call them. This means that values and functions that exist in the IMS interface must coincide correctly in the right sequence and of the right type. In testing a particular module i.e. Stock Type module, data must be entered first for a new stock type before it can be saved based on the user interface whereby the input boxes appear first before the 'Save' button.

7.2.4 White Box Testing Versus Black Box Testing

White box testing can indicate test considerations that are not produced by black box testing and vice versa.

White box testing is concerned only with testing the software product; it cannot guarantee that the complete specification has been implemented. Black box testing is concerned only with testing the specification, it cannot guarantee that all parts of the implementation have been tested. Thus black box testing is testing against the specification in IMS and will discover *faults of omission*, indicating that part of the specification has not been fulfilled. White box testing is testing against the implementation of IMS and will discover *faults of commission*, indicating that part of the implementation is faulty. Therefore, in order to fully test the IMS both black and white box testing are required.

White box testing is much more expensive than black box testing. It requires the source code to be produced before the tests can be planned and is much more laborious when trying to determine the suitable input data and to verify if the software is or is not correct. Therefore when testing the IMS, black box testing was started as soon as the specification is available. White box test commence as soon as all black box tests have been successfully passed, with the production of flow graphs and determination of paths. The paths are then checked against the black box test plan.

7.2.5 Unit test on the IMS

Below is an example of unit testing on the Receipt Module. The unit testing on all the modules in the IMS can be viewed in the Appendix.

Module : Receipt

No	Test Procedure	Error / Output
1.	Role clerk : Click Item ID/Description	Item clicked
2.	Click on Item in Stock Listing form	Stock listing clicked
3.	Enter the quantity to be sold	Quantity entered
4.	Click 'save'	Stock saved
5.	Select item on the list below	Item selected
6.	Click 'delete'	Item deleted
7.	Click 'save and print'	Receipt saved and printed
8.	Role : Admin and Supervisor Click '...'button	Receipt listing pops out

Figure 7.2 Unit test for Receipt Module

7.3 Integration Testing of IMS

The purpose of integration testing is test sufficiently and adequately whether or not the IMS is able to run as one whole program. This is to ensure that all the modules within the IMS are able to function properly with each other and run as one complete system. Below are the methods of integration testing conducted on the IMS.

7.3.1 Big – Bang Integration

The big bang method is the most popular among all the methods of integration testing.

This is where the modules are chosen and tested randomly against each other to see if they are able to function as a complete system

The main modules i.e. Stock Type, User Authentication, Stock, Chain Store Information, Report, Shortage Notification, Receipts and Help are first tested as individual modules (unit testing) before proceeding with the big bang integration test.

The Stock Type module is then tested against the Stock Module. Data that was entered in the Stock Type modules (i.e. a new stock type is entered and its input values are Stock ID, Stock name, weight and its unit in weight (kg or g), and cost and sales price) must appear when the Stock ID is selected in the Stock module. The data that appears in the Stock module must coincide with the data in the Stock Type module.

The same theory applies to the Receipt modules. When a stock type is selected for sales, the current and critical quantity in the Receipt module must be exactly the same as the data in the Stock module. When the sale of a stock occurs, the current quantity would then lessen in the Receipt Module. This quantity must be verified against the quantity of that particular stock type in the Stock Module.

However, the Big – Bang Integration test does have its drawbacks. One major problem is that it tends to uncover a large number of bugs at once stating what situation caused that error to occur.

Another setback is that once errors that occur between 2 modules are corrected, if a 3rd module is integrated, then this generates more errors between all three of the modules.

Therefore although the Big – Bang Integration test is a popular choice, it is best to test the system with other integration test methods to confirm the functionality of the system.

7.3.2 Top – Down Integration

This method of integration testing implies that the top level of the IMS is tested first and then stubs are added gradually till lowest level of the system.

The Reports module can be likened as the top level of the IMS System. Therefore after unit testing is conducted on the Reports module to ensure that the module itself is functioning properly, it is then tested against Receipt module. The Receipt module is one

level lower than the Reports module. The sales and transaction that occur at the Receipt module level affects the Reports module. Hence, the data in the Reports module such as sales, stock quantity left is verified against the data in the Receipt module. The two modules must be able to function efficiently and correctly against each other.

The Receipts module is dependant on the Stock module that is one level below. The quantity of the stocks sold is based on the quantity of that stock in the Stock module. The data in Receipt Module must tally with the data in the Stock module.

Top down integration testing verified that the three modules that are the Reports, Receipt and Stock type module are able to function properly with each other.

7.4 System Testing

System testing is a series of varied tests that is designed to fully and thoroughly exercise the system to uncover its imperfections and to measure its capabilities. The main purpose of this is to test the integrated system in order to verify that it has met the specified requirements.

7.4.1 System Test Consideration

This particular testing method, not only the behavior of the individual functions is tested but further functional tests such as the ones below are needed.

7.4.1.1 The Event List

In the event list, all possible triggers of events in the IMS must be conducted and the results obtained are then compared with the expected results. This is done by testing every function with one or more events in the event list. The functions are tested as part of the system and as part of the user interface.

For the IMS, various inputs and actions were conducted to verify if the results tally with the expected outcome. For example, the information the Stock Type module should trigger the information in the Stock Module and not the other way around. Tests were also conducted on the Shortage Notification module to determine that the notification would be generated only when the stock is below the critical quantity. This is verified after stringent events list test was conducted on the system.

7.4.1.2 Specific Scenarios

Specific situations and user profiles were created for the IMS and tested against the system. Scenarios help to answer questions on the system requirements and its limitations.

Examples of scenarios conducted are shown below.

Scenario	Actual Result	Expected Outcome
User enters with the role Clerk.		
1. What are the modules accessible by the clerk?	Receipt Module	Receipt Module
2. Are they able to access other chain stores?	No. They have access to their chain store by default.	No. Access to their chain store is by default upon login.
3. Can the clerks change values i.e. Sales Price?	No. The values are displayed but disabled.	No. The sales price is determined only by the Administrator.
Scenario	Actual Result	Expected Outcome
User enters with the role Supervisor.		
1. Can the Supervisor view other Chain Stores details?	No. They are able to view only their own chain store.	No. Access is denied to other chain stores
2. Are they able to add stocks for other chain stores?	No. They can only do so for their respective chain store.	No. Access for adding stock is limited to their respective chains tore.

Figure 7.3 Scenario for System Test

7.4.1.3 Error Message Testing

All error messages and handling are tested for their appropriateness and understandability from the users point of view.

Error messages in the IMS such as "Logon to Print Server Failed! Please consult the administrator" when reports are to be generated is checked to determine if the user understands the error generated and is able to react to it appropriately.

Error messages in the IMS are written in simple and polite manner as to not confuse and create panic when using the system.

7.4.1.4 Screen Mapping

All screens in the IMS including the sub menus and pull -down menus are tested to ensure that when clicked or on-mouse, is connected to correct screen.

When the Reports module is clicked in the main screen, the sub menu of Sales of Chain Stores and Number of Stocks left appear as stated in the requirements.

Screens in the IMS are designed in a similar style and placed in a manner that makes it easy to locate them.

7.4.1.5 Documentation Testing

Examples and steps in the IMS user manual are tested for correctness and to verify if the manual is able to answer any questions or doubts posed by the user. Terminology used in the user manual must be understood and comprehended by the users.

7.4.2 Fundamental Tests (Product Verification Testing)

7.4.2.1 Usability

The IMS system is built with consideration for ease of use. This is done by building the user interface that contains patterns that are already familiar to the typical user. The users of the IMS system must be able to use the system easily without having to depend on the user manual.

7.4.2.2 Reliability

Reliability testing is conducted to determine that when the same input values are entered into the system, the output values must be the same. The IMS system is tested to ensure that the system is able to operate for a period of time before a fault is detected. Reliability is important in establishing that the IMS system does not fail when being used.

7.4.2.3 Installability

This particular test ensures that the IMS must be easy to install even by a layperson correctly and easily without the help of an expert.

7.4.2.4 Performance

Performance test was conducted on the IMS to ensure that the response time is able to meet the users requirement and does not exceed the specified performance criteria under heavy data volume or stress. For example, the Shortage Notification in the IMS system must be able to display the stocks below the critical level within a 5 second time span when the administrator or supervisor logs into the system.

7.5 Test Case for IMS

Test cases are important in creating a scenario and then testing the system against that scenario to determine the error generated.

Test cases for the IMS are included in the Appendix.

7.6 Conclusion

A system that is not tested can be likened to driving a car that was assembled within a day; you do not know when, how or why a system has failed.

Proper testing methods and approaches are compulsory before a system is marketed. In fact a system is not complete without testing. Therefore, it is important to conduct tests at all levels of the IMS system such as unit testing, integration testing and system testing to ensure that the system is complete, correct and meets the users requirements.

Chapter 8 – System Evaluation

System evaluation is basically assessing and reviewing the IMS system from the time of conception until it is tested and released.

The main purpose of system evaluation is to establish that the IMS system requirements have been met while establishing the system's assets and strengths. System evaluation also uncovers the limitations of the IMS as well as discussing the future enhancements to be included into the system.

8.1 IMS System Strengths

The IMS System was designed with several benefits in mind, including the ones stated below.

8.1.1 Ease of control and manipulation

One of the main advantages of the IMS system is that it is easy to use and operate. The functions displayed are placed strategically and short cut buttons are included so that users can access the system effortlessly.

8.1.2 User – friendly Interfaces

The interfaces of the system were designed with the users expectations in mind; that is the interfaces must be easy to view and enter information without overcrowding of data. Minimal graphics are displayed as too much would only distract and confuse the user. Modules are designed with the similar pattern to maintain consistency and similarity.

8.1.3 Database Transparency

One of the plus points of the IMS system is that the users need only to be concerned with the data that they enter or retrieve within the application. The system itself will retrieve or update the data to the database automatically; users need not know the format used in storing the data and the physical database structure.

8.1.4 Security

One of the most important characteristics of the IMS system is its high security level. Users that log into the system are automatically defined by their user role, i.e. Administrator, Supervisor or Clerk. The modules that are displayed to the users are based on their access level to the system. Therefore, user intrusion or unauthorised access to modules is prevented, maintaining the integrity of the IMS system.

8.1.5 Maintenance

The IMS system is able to function accurately with minimal maintenance. It does not require constant user attention on the maintenance of the codes as they are designed to be dynamic. Adding new functions would not disrupt the existing modules severely.

8.2 IMS System Limitations

Although the IMS system was designed to the best of abilities, it does however have a few restrictions such as listed below.

8.2.1 Speed of data retrieval

At times, data retrieval especially reports generation might be slow and require longer processing time. This is because connecting to the Crystal Reports windows requires a slightly longer time instance of 6 seconds to generate a report for the first time.

8.2.2 Predefined User Roles

In the IMS system, there are 3 predefined roles that are Administrator, Supervisor and Clerk whose access to the modules are predetermined by the system. Therefore users who are not classified into either one of these 3 roles cannot access the IMS system.

8.3 IMS System Enhancements

Although the IMS system has fulfilled its requirements, there is still room for future enhancements and development of the system as listed below.

8.3.1 Include Stocks Supplier

As for now, the scope of the IMS system is extended to only the administration at the main office and the chain stores. However in the future, the supplier would be included to extend the scope of the system. By including the supplier, the administration can determine where and from whom to buy a particular stock type. The administration can also determine which supplier provides the stocks faster and efficiently to the chain stores.

8.3.2 Integrating the IMS as part of Teaching Module

The IMS, although would take a long time to become a fully functional inventory management system in a real world situation, could be used as a teaching tool for new users of the system such as clerks and supervisors who might not be computer literate. The IMS which has an easy to use and manipulate user interface, would present the new users a better understanding of the inventory system

8.3.3 E - Mail module

An e-mail module would be included into the IMS system to better extend the communication between the administration at the main office and the chain stores such as notifying the administration of the absence of staffs or other upcoming issues.

8.4 Problems Encountered

8.4.1 Not familiar with the programming language

Among the difficulties faced when developing the IMS system was lack of programming skills particularly in Crystal Reports programming in producing graphs based on the reports generated.

Solution: Online referencing and related books proved to be very helpful in determining not only the correct way to generate graphs but also in determining the suitable graph type to choose when generating the reports.

8.4.2 Difficulty in obtaining end user information

Minor problems were encountered in trying to obtain information from the chain stores either from time constraint or rules and regulations of the selected chain stores.

Solution : Conduct brief and informal interviews while observing the operations of the chain store as to not disrupt the schedule of the supervisors and clerks at the chain stores.

8.4.3 Abundance of information in the Internet

Although the Internet has a vast source of information on inventory management systems, there was difficulty in filtering the information to select the ones most related to the system developed.

Solution: Read as many related articles as possible and stream the important and related facts by summarizing them.

8.5 Knowledge Gained

8.5.1 Knowledge on Additional Software Tools

Knowledge and skills on software tools such as Microsoft SQL Server and Crystal Reports was gained especially in learning the correct syntax to connect one software tool to another.

8.5.2 Good Graphical User Interface Design

Although at times, graphical user interfaces are considered less important than the coding within the system, it is still important to develop good user interface design skills. This is because end user interaction depends on the design of the user interface rather than the coding itself. Therefore, good graphical interfaces will encourage the users to utilize the IMS system.

8.5.3 Skills in gathering information and fact

Developing the IMS system helped to cultivate skills in gathering relevant and useful information such as data through interviews and articles. This information was then used in designing the IMS system.

8.5.4 Experience in Problem Solving

Perhaps the most important knowledge gained through out the development of IMS was the ability to solve problems that arose. The ability to think out of the box helped to solve problems relating to user requirements and coding.

8.5.5 Learning to work independently

The ability to research and make decisions without depending wholly on others proved to be a worthy skill. This taught self – sufficiency and not to rely solely on one particular resource, rather gather information from various sources and analyse them thoroughly.

8.5.6 Skills in writing documentation

A system is never completed without proper documentation. Therefore, writing a documentation that is complete, comprehensive and systematic is important in relaying information to the user about the systems' requirements and functionality.

8.6 Conclusion

The IMS system was assessed thoroughly from different angles to establish its benefits and also its limitations. System evaluation helped to determine to the future enhancements to the system as well as knowledge and experience gained throughout the development of the system.

Bibliography

1. Dixon, R. 1995 *"Client/server and Open System, Technologies and the Tolls That Make Them Work"* John Wiley & Sons, New York
2. Burke, D., Calabria, J. 2000, *"Teach Yourself Lotus Notes and Domino R5 Development in 21 Days"* SAMS, Indianapolis
3. Powell, T.A., Whitworth, D. 1998, *"HTML"* McGraw – Hill
4. Pfleeger, S.L. 1998, *"Software Engineering Theory and Practice"* Prentice – Hall
5. Davis, A.M. 1993 *"Software Requirements: Objects, Functions and States"*
Prentice – Hall
6. Kendall, K.E., Kendall, J.E. 1999, *"Systems Analysis and Design"* Prentice – Hall
7. Weinschenk, J., Jamar, P., Yeo, S.C., 1997, *"GUI Design Essentials"* John Wiley & Sons, New York
8. Jarvis, A., Crandall, V. 1997, *"Inroads to Software Quality"* Prentice Hall

URL

1. <http://www.vic.edu/depts/acc/hardware/notes/domino.html>
2. <http://www.sperd.net/trc/lotustraining/>
3. <http://www.barcode-system.com>
4. <http://itdimensions.com/support/resources/databasaeoption.html>
5. <http://www.uclan.ac.uk/facs/destech/compute/staff/casey/integ/waterf.htm>
6. <http://www.adobe.com>
7. <http://www.lotusadvisor.com>
8. <http://www.vbtutor.net/lesson1.html>
9. <http://www.jtap.ac.uk/admin/dissemination/petch/tsld019.htm>
10. <http://www.cs.toronto.edu/~sme/CSC444F/slides/L04-Lifecycles.pdf>
11. <http://www.jtap.ac.uk/admin/dissemination/petch/sld019.htm>
12. <http://www.crystaluser.com>
13. <http://www.crystaldecisions.com/products/crystalreports/default.asp>
14. <http://www.dominopower.com/issues/issue200006/crystal0600001.html>
15. http://www.ilearn.to/amsu/course_cat/amsu_advanceditoffice.htm
16. <http://www.oracle.com>
17. <http://www.islandnet.com/~tmc/html/articles/orareln.htm>
18. <http://www.microsoft.com/sql/default.asp>
19. <http://stylusinc.com/website/microsoftSQL2000.htm>
20. http://www.manageddedicatedservers.com/microsoft_sql_server/microsoft_sql_server.html