OSEI TUTU II INSTITUTE FOR ADVANCED ICT STUDIES
GHANA

AFFILIATED TO

KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY, KUMASI

SCHOOL OF GRADUATE STUDIES

THE DEVELOPMENT OF SOFTWARE FOR MAINTENANCE MANAGEMENT IN THE HOSPITALITY INDUSTRY

(Case Study Golden Tulip Kumasi City Hotel)

A DISSERTATION PRESENTED TO THE OSEI TUTU II INSTITUTE FOR ADVANCED ICT STUDIES IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE AWARD OF THE DEGREE OF
MASTER OF SCIENCE IN ADVANCED ICT STUDIES
(MANAGEMENT OPTION)

BY
STEPHEN KWADWO OSEI
OCTOBER, 2009

DECLARATION

I hereby declare that this study was under taken independently and it is my original work.

It is not replication of any work either published or unpublished. All references made in this study are dully acknowledged. Finally, all aspects of this study have been discussed with and approved by my supervisor, Prof Vasko Fournadjiev

Signature ..... Date 21/10/2009

STEPHEN KWADWO OSEI
(STUDENT)

I declare that this dissertation was written under my supervision and that the Student has been consistent in his interaction with me for guidance and direction. He has my consent to present it for assessment.

Signature ..... Date .....

PROF. VASKO FORNADJIEV

(SUPERVISOR)

DEDICATION

This work is dedicated to Mr. Osei Frimpong, Ms. Agnes Boateng, Yvonne, Helen and Madam Cecilia Duku.

To MJ.

ACKNOWLEDGEMENTS

I express my sincere gratitude to Prof. Vasko Fournadjiev who has been a great mentor through the two intense years spent at the Institute and who whole-heartedly supervised my project work. God bless you richly for your immense help.

Many thanks to Prof. Martin Looijen, the rector of the Institute, for his guidance, and the entire staff for their support.

To Mrs. Sophia Quashie-Sam for understanding our plight and standing by us the entire time.

To Harmien Dam, whose unique sense of humour cheered us up when the going got tough.

To the General Manger and staff of the Golden Tulip Kumasi City hotel for their warm reception during my internship.

I would also like to thank Mr. Emmanuel Afful , Godfred Adzokatse and MT for their immense assistance and encouragement. Emma, thanks a million!

Special thanks to Mr. Osei Frimpong, Agnes Boateng and my grandmother, Madam Cecilia Duku for their love and support through the years. To my siblings Yvonne and Helen, I say, thanks a lot.

To the Ultimate Group, for coming up with the ultimate ideas and putting us on top.

To God Almighty.

STEPHEN K. OSEI

OCTOBER 2009

ABSTRACT

The objective of this study was to develop a maintenance management information system for the hospitality industry. The Golden Tulip Kumasi City Hotel was used as a case study.

The software consists of an asset classification module which places assets into a tree structure for easy identification. Along with this module is a room number generator and locator for the assets.

The software also catalogues a list of staff capable of maintenance services together with their working shift time table.

At the core of the software are the planned maintenance module, which has the responsibility of automatically processing preventative maintenance routines, and the unplanned maintenance module with which staff can report of assets that need maintenance outside regular schedules. The software then suggests a list of capable staff from which the maintenance administrator can choose from. If desired, the software is also able to automatically dispatch work orders to staff based on fair distribution of work load.

The final module tracks the history of the assets, which enables management to make informed decisions on the assets.

The project uses client-server architecture and is designed to work in a computer network environment.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CMMS: Computerised Maintenance Management System

SQL: Structured Query Language

DBMS: Database Management System

RDBMS: Relational Database Management System

GUI: Graphical User Interface

RAD: Rapid Application Development

OLEDB: Object Linking and Embedding Database

TCP/IP: Transmission Control Protocol /Internet Protocol

PMU: Planned Maintenance Unit

MM: Maintenance Manager

SUV: Sports Utility Vehicle

VW: Volkswagen

ID: Identity

PDF: Portable Document File

1 INTRODUCTION

1.1 Background

Every organization has assets it employs for its day to day activities. To stay in good running condition and to prevent incurring heavy cost of repairs, some of these assets need to be maintained periodically. The hospitality industry forms a group of such organizations that requires regular maintenance of assets to be able to provide top class services with minimum interruptions.

1.2 Problem Definition

The Golden Tulip Kumasi City Hotel is a relatively new hotel that was established in January 2008. The hotel has made heavy investments in a lot of equipment. Most of these equipment need to be serviced periodically. There are also lots of requests for maintenance that are not planned. For these reasons, it has become extremely difficult for the hotel to track down all maintenance activities and to attend to them on time. There is also the need to track the associated cost of maintenance and to isolate problematic equipment categories for the hotel's management to make informed cost decisions.

For these reasons, this thesis aims at developing software to manage the maintenance activities.

1.3 Objective of Study

The objective of this study is to develop an information system for maintenance management for the hospitality industry.

The system should have the following functionalities:

- Categorizing assets for easy identification,
- Cataloging a list of staff capable of maintenance services together with their working shift time table,
- Generating automatic maintenance schedules,
- Receiving request for unplanned maintenance from remote ends,

1.4 Justification

This thesis aims at developing an information system for managing maintenance activities in the hospitality industry. The Golden Tulip Kumasi City Hotel was taken as a case study. This was in view of the fact that there was no computerized maintenance management system in place. Request and scheduling activities in the maintenance department which could be made easier with a computerized information system were done manually and thereby were prone to human errors and slow response times.

The benefits of an information system for maintenance management include but not limited to:

- Reduction of equipment downtime by the ability to promptly report faults.
- Minimization of the frequency of equipment breakdowns by enhancing preventative maintenance scheduling.

- Reduction of maintenance cost due to minimized equipment breakdowns.

1.5 Methodology Used

The methodology used comprises:

Literature Review

- This part involves the review of literature on maintenance management
- It also involves finding relevant articles, technical documentation and papers on approaches for developing software applications.

Interviews

- Interviews were conducted with the hotel staff for their requirements.

Requirement Analysis and Software design encompasses:

- Understanding what is required of the system
- Consideration of which software development architecture to employ
- Choice of programming environment and database
- Program flow
- Database design

Software Coding

- This stage involves the writing of software code.

Testing and Evaluation

- Sample case studies are fed into the system to ascertain if the correct functionality and results are obtained
- Debugging will also be done at this stage.

1.6 Scope

This thesis focuses on the technical development aspect and does not extend to the soft managerial issues.

1.7 Limitations

The staff Work-Shift (time table) assumes a single shift per staff per day and does not cover multiples shifts per staff in a day.

The Work-Plan is only drawn for planned schedules since problems in unplanned maintenance activities may not be known in advance.

1.8 Thesis Outline

Chapter 1 provides an introduction and overview of this document. Chapter 2 delves into maintenance management and associated software used in maintenance activities as well as the review of some of the tools that are used for the design of the software. Chapter 3 gives the maintenance processes in existence at the hotel and tries to determine which ones are automatable and which ones are not. Chapter 4 discusses the architecture used in the thesis. Chapter 5 deals with the database and software components of the system. Chapter 6 deals with the design and implementation of the system. Finally, Chapter 7 concludes the thesis.

2 LITERATURE REVIEW

2.1 Management

Management can be defined as the art of handling or controlling something successfully or the skilful handling or use of resources. (Encarta, 2009). It entails planning, allocation of resources to the plan and evaluation to see if the end result(s) is/are in line with what was expected (Shead, 2007). It can be inferred that the key element in management is coordinating all components within a given system for a desired outcome.

A set of processes and procedures that an institution or organisation uses to achieve such an outcome forms what is known as a management system (Wikipedia, 2009).

2.2 Maintenance

Maintenance can be defined as actions that are required to keep a machine or system in service (Wiktionary, 2009). This could include but not limited to, replacement of worn out parts, lubrication, cleaning, as well as other servicing performed on assets.

Putting these two definitions together, we could propose that Maintenance Management is the skilful handling of the actions that are required to keep a machine or system in service.

2.3 Maintenance Management System

Maintenance management deals with "the process of leadership, organizing, planning and scheduling" of preventative maintenance, condition monitoring, and execution of maintenance repairs and recording of technical data in support of management (IDCON, 2009). A Maintenance Management system is, therefore, one that is designed in support of the above. When such a system is aided by computer software, it is known as Computerised Maintenance Management System (CMMS).

The CMMS stores an information base of the maintenance activities undertaken by an organisation. The stored information helps in classification and easy location of asset for easy identification and application of maintenance schedules as well as helping management take reasonably informed decisions (Wikipedia, 2009).

CMMS may be specific to the maintenance operations of an organisation or can be generic serving the needs of several organisational types. For example, the needs of a health care organisation may be different from that of the hospitality industry. They could also have some routines in common. CMMS may include one or more of the following (Wikipedia, 2009):

- Management of Asset: Consists of keeping records of assets. Such records may contain information pertaining to warranty, service contracts, life expectancy of asset, dates of purchase. Generally, it keeps information that may be relevant to management and maintenance work.
- Orders for Work: It comprises but not limited to, scheduling of jobs, dispatching of staff to the job, and documenting the cost of repairs/service.

- Inventory Management: Includes management of materials, equipment and replacement parts, determining restock level and order placement for new stock.

2.4 Review of Some Software Systems for Maintenance Management

2.4.1 Bigfoot CMMS

Bigfoot CMMS is designed by Smartware Group (Smartware Group, 2008). It is a multi-sector computerised maintenance management system designed for the healthcare, hospitality, education, financial, food and manufacturing industries.

Bigfoot comes in two versions: The 'Traditional' version and the 'Internet' version.

The 'Traditional' version is ideal for small facilities where the software database is installed on a local server and access to the internet is minimal or nonexistent. It is faster than the 'Internet' version.

The 'Internet' version is ideal for corporations with several facilities whose maintenance activities need to be managed. This version provides access to the system from virtually any place where there is internet availability (Smartware Group, 2008).

Bigfoot presents a generic solution for maintenance management and does not focus specifically on the hospitality industry.

2.4.2 CWorks CMMS

CWorks maintenance management system is designed by CWorks Systems Berhad. It is a CMMS for managing the maintenance activities of medium sized organisation. It has an asset module for cataloguing assets, a location module for locating asset, work request module for receiving work requests, a work scheduling module, employee

information module and a reports module for generating reports. CWorks is built on a Microsoft access database and as such not too well suited for enterprise-wide use (CWORKS SYSTEMS, 2009). It lacks support for managing the time-tables of employees.

2.4.3 Clever CMMS

Clever CMMS is another CMMS with support for maintenance scheduling, asset management, report generation. It is designed by Clever Solutions. It has a user friendly interface and supports multiple concurrent users. It also has an inbuilt messaging system for intercommunications between users (Clever Solutions, 2009).

Clever CMMS is also a generic system that does not specialize in the specific needs of the hospitality industry.

2.5 Client-Server Architecture

A server can be defined as an application that provides a service that is requested for by another application. A server generally sits and waits for a request before responding to that request. A client on the other hand, request for a service from a server. It is the consumer of the service (Ahmed, Garrett, & Faircloth, 2002)

A Client-Server Architecture separates the presentation layer (user interface) and data access layer (database) by running them logically or physically separate platforms. The business logic may be kept on either of the two platforms or can be on its own platform.

Client-Server applications can also be referred to as multi-tier or n-tier applications. The simplest of multi-tier applications is the two tier application. Figure 2.1 shows a diagrammatic representation of a simple two-tier application. Some of the advantages of a multi-tier application include (Sempf, 2005):

- Replacing one component does not require changing the rest (modular design)
- Replacing one component does not affect the functionality of the rest (modular design)
- Placing each component separately enhances:
    - Security: Since access to one component does not guarantee access to the others.
    - Reduced Load: If a client is involved in some processing, such load is taken off the server.
    - Availability: As a result of the reduced load where each client is involved in some processing, there is minimal congestion and this contributes to the availability of the service.(An exception is when the server does all the processing and pushes the result out to the client)

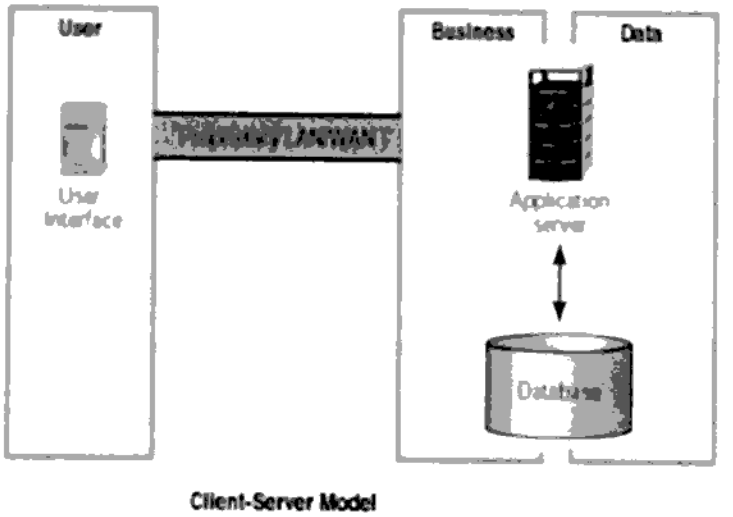


Figure 2.1 Two-tier client server model

The stated advantages above makes client-server architecture well suited for the development of a maintenance management system. Such a maintenance management system will take requests from staff at different sites for the review of the maintenance administrator.

2.6 Relational Databases and Relational Database Management System

A database can be defined as a structure for storing data which is arranged for efficient and fast retrieval (Petroutsos & Bilgin, 2002). A database refers to a structured organised group of records (Price, 2003).It is a repository of information (McManus & Goldstein, 2003). The data they contain can be queried in many ways for information retrieval. Databases are essential for a multi-tier (client-server) system (Bryla, 2003). They provide the backend to maintenance management applications and indeed, most information systems related applications. A relational database is a database which groups related data into tables and defines logical links (relationships) between the tables (Kauffman, Claudio, Chinnathampi (Das), Greenwood, & Matsik, 2002). It uses two-dimensional tables (relations) to store information for supporting a business (Bryla, 2003).Figure 2.2 shows two tables with a relationship defined.

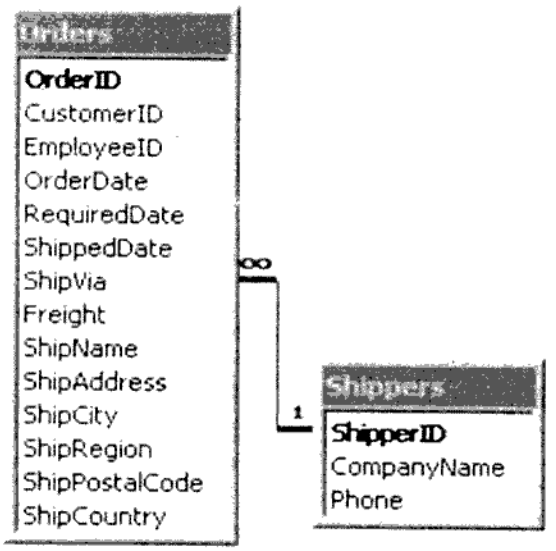


Figure 2.2 Tables with a relationship defined

Advantages of using a relational database include (Buzzle.com, 2008):

- Support of Normalization: Normalization refers to the step by step process of removing redundancies(repetitions) from a database and ensures that the database is free from undesirable characteristic relating to the update, deletion or addition of data (Wkipedia, 2009)
- Support of Structured Query Language (SQL): SQL a database language that is easy to understand and with a syntax that is human readable. SQL is also supported by many vendors
- Support for Dynamic Views: A view is a snapshot of data that can comprise data from several tables. A relational database view is not static and can change on-the-fly as the underlying data changes. Views are used to simplify complex data by presenting only the entries that are needed.
- Support for Security: Relational databases provide the functionality of limiting access rights.
- Relational databases are scalable and well suited for the implementation of a Multi-tier(client-server) application.

A Database Management System (DBMS) controls the creation, utilization and maintenance of databases. A DBMS designed to work on relational databases is known as Relational Database Management System (RDBMS) (Price, 2003).

2.7 Software Development Process

Software development process refers to a guideline structure that is undertaken for the development of software (Wikipedia, 2009) .

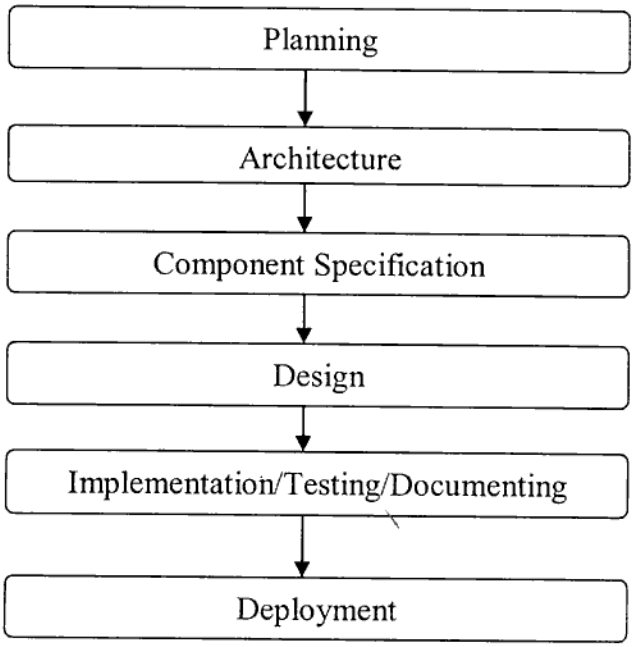


Figure 2.3 Software Development Process

The activities involved in the Software Development Process, also known as the Software Life Cycle are (Wikipedia, 2009) :

- Planning: This involves the gathering of requirements from end users and analysing those requirements. Scope of the project is defined at this point.
- Architecture: This refers to an abstract representation of the system. It aims at ensuring that the software will meet its laid down requirements.
- Component Specification: This involves a precise description of the software components and functionalities to be written

- Design: This stage encompasses the logical design of both the database and interfaces used in the system.
- Implementation, Testing and Documenting:
    - Writing of software code encompasses the implementation stage.
    - Testing is done to identify bugs for correction
    - Documentation of the design of the software is crucial for the future maintenance of the software.

2.8 Software Programming Languages

A programming language is a form of non-human language designed to enable machines perform specific tasks. As pertains to computers, a programming language can be defined as a language used to instruct computers to perform a task of computation or algorithm as well as controlling other devices. (Wikipedia.org, 2009)

2.8.1 CodeGear Delphi

This programming language is developed by CodeGear. It has its roots in the Pascal programming language. Delphi is well suited for desktop and enterprise database applications. It is relatively easy to use and stable (Wikipedia, 2009).

2.8.2 Visual Basic 2005

Visual Basic 2005 is developed my Microsoft. It has its roots in the BASIC programming language. It is a Rapid Application Development (RAD) tool and has rich Graphical User Interface controls (GUI). The most distinguishing feature of all BASIC programming languages is that they are easy to read and understand and follow an

English-like syntax. Visual Basic is robust and stable. The Express Edition is also available for free (Wikipedia, 2009).

2.9 Database Management Systems:

2.9.1 Microsoft Access

Access is an RDBMS developed by Microsoft and is part of the Microsoft Office application package. It is based on the Microsoft Jet OLEDB engine. Microsoft Access was developed for the desktop environment. It is relatively easy to install and administer. It is also cheaper than most RDBMS software. Access is not well suited for a Client-Server environment. It supports a limited number of concurrent users and does not scale very well as the amount of data grows to enterprise-level sizes (Dewson, 2006).

2.9.2 Oracle

This is a relational database management system developed by Oracle Corporation. This RDBMS has a high user base and is used extensively by search engines. Oracle is a very powerful and stable. It is highly scalable and fast. It offers flexibility by providing many add in tools. It is well suited for the Client-Server environment. Oracle is also noted to be quite complex to administer and expensive (Dewson, 2006).

2.9.3 Microsoft SQL Server 2005

SQL Server 2005 is developed by Microsoft Corporation. This RDBMS is stable, good for simple applications as well as scales well with large (enterprise) applications. SQL Server 2005 is relatively inexpensive. The Express edition is available for free. It is user-friendly and provides many tools for analyzing data. It is also relatively easy to install

and administer. The Express edition does not come with many of the analysis tools as the commercial versions and is not intended for very large commercial applications (Dewson, 2006).

2.9.4 Sybase

Sybase is a very fast and robust relational database management system. It is known to be rebooted, on average, only once or twice in a year. Sybase is predominantly used in the Unix operating system environment. It is administered mainly by command-line and is not very user friendly. Sybase has very limited Graphical User Interface (GUI) (Dewson, 2006).

3 PLANNING

3.1 Overview of Existing Maintenance Processes

The preliminary work on specification gathering for the software was done on site by observing the flow of work at the maintenance department of the Golden Tulip hotel. These processes are outlined in Figure 3.1

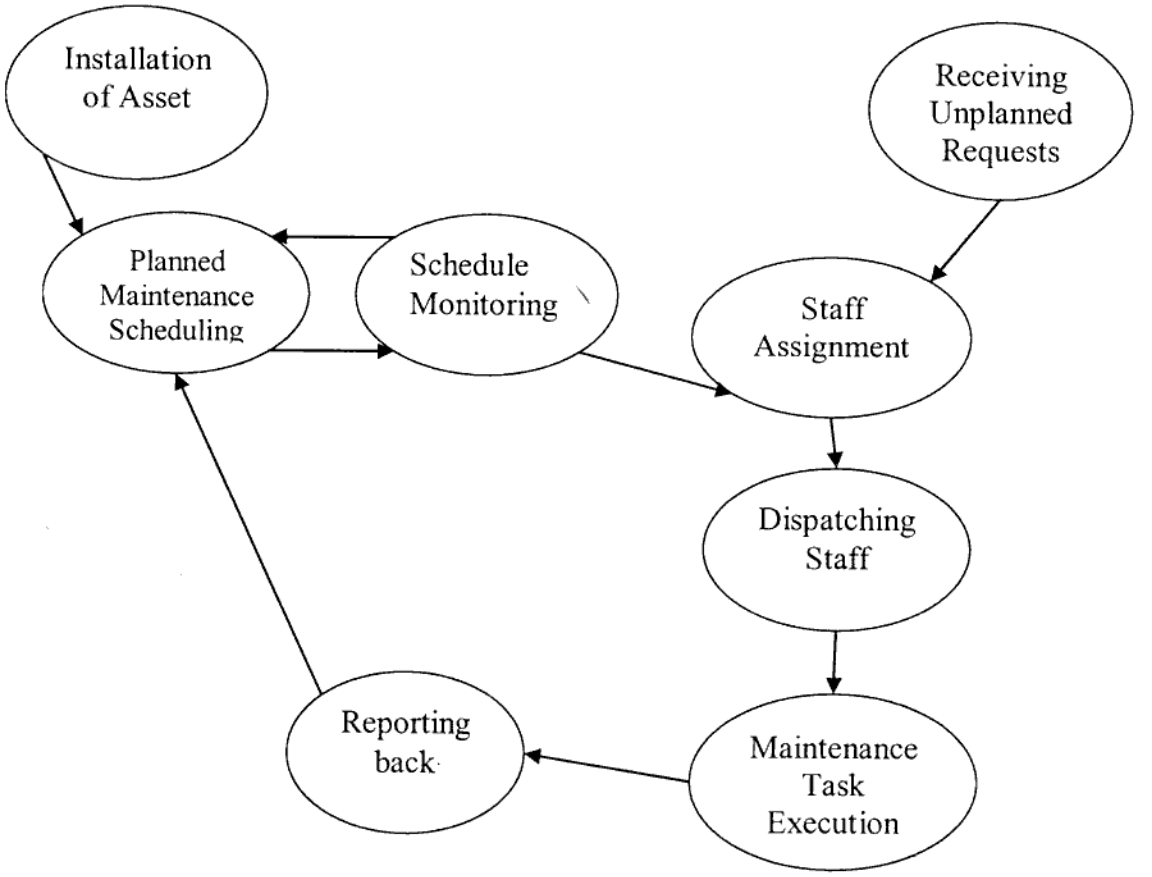


Figure 3.1 Outline of Existing Maintenance Processes

3.1.1 Installation of Assets

Actors: *Technical Staff, Maintenance Administrator.*

Input*: asset data, location data*

Output: *asset catalogue*

Typical steps for a new installation are as follows:

- A new asset (equipment) is catalogued and given an asset identification number (Asset ID). The catalogued information includes that specified by the manufacturer for its preventative maintenance.
- A technician with the requisite skills for installation is assigned and the equipment is taken to its designated location for installation.

3.1.2 Planned Maintenance Scheduling

Actors: *Maintenance Administrator*

Input: *Asset data*

Output: *maintenance schedule*

The following are the steps involved in creating a preventative (planned) maintenance schedules:

- Information from the manufacturer on the preventative maintenance interval is recorded in the cataloguing phase of the installation of the asset.
- If such information does not exist, the maintenance engineer/administrator together with a technician may provide such information based on experience with such equipment or based on discretion.
- A list of the recommended preventative maintenance information is created and printed out.

3.1.3 Schedule Monitoring

Actors: *Maintenance Administrator, Technical Staff*

Input: *maintenance schedule*

Output: *monitor schedule*

Monitoring of the planned maintenance schedule entails the following:

- Technical staff are given a copy of the schedule list and assigned to some of the equipment.
- The technical staff are supposed to go through the list on a daily basis and attend to the eequipment that need servicing.
- The maintenance engineer/administrator goes through the list and prompts the technical staff on their schedule.

3.1.4 Staff Assignment

Actors: *Maintenance Administrator*

Input: *staff skill data, service skill data, staff work load*

Output: *staff assignment*

Technical staff are assigned to schedules or tasks via the following procedure:

- Information on the technical capabilities of the staff is stored in a file
- The information on the asset that is due for maintenance is checked for the kind of skill that is required for the service.
- Based on skill required, a member of the technical staff is selected. This is also based on the work load on the staff.

3.1.5 Staff Dispatch

Actors: *Maintenance Administrator, Technical Staff*

Input: *maintenance schedule, staff assignment*

Output: *Staff Dispatch, Dispatch form*

- The maintenance administrator issues out a dispatch form to the staff assigned to the task. On the form are the location of the asset, the date of dispatch and the name of the asset.

3.1.6 Maintenance Task Execution

Actors: *Technical Staff, Maintenance Administrator/Supervisor*

Input: *Dispatch form, Staff Dispatch*

Output: *Physical Execution of task*

- The technical staff physically executes the task and reports back to the maintenance administrator
- The maintenance administrator/supervisor then inspects the work

3.1.7 Reporting Back

Actors: Maintenance Administrator, Technical Staff

Input: *Dispatch form, Schedule Data*

Output: *maintenance schedule*

- After inspection of the task, the dispatch list is signed by the maintenance administrator.
- The maintenance schedule of the asset is then updated.

3.1.8 Unplanned Maintenance Requests

Actors: *General staff, Guests*

Input: *Asset data*

Output: *Request Maintenance*

When a piece equipment breaks down, the owning department sends written or verbal request to the maintenance engineer/administrator. The administrator then assigns a technician with the requisite skills to the task. After execution, the technician reports back. The administrator then inspects the work. The administrator then informs the owning department of the outcome.

3.2 Automatable Processes

There exists the possibility of automating some of the maintenance processes or activities. Some processes can be fully automated and others partially. The under listed gives an outline of processes and activities that can be automated and those that cannot..

3.2.1 Installation of Assets

Level of Automation: *Partial*

The physical installation of an asset cannot be automated with maintenance management software. It requires human discretion and expertise. Installation procedure also varies from one asset to the other. However, cataloguing of the asset can be fully automated since it is primarily a record keeping activity. Selection of staff with the requisite skills can be automated since both the details of staff and the skill required can be electronically captured.

3.2.2 Planned Maintenance Scheduling

Level of Automation: *Partial*

The gathering of the initial information that is needed for scheduling cannot be automated. It requires human effort and discretion to gather this information, especially, when not stated by the manufacturer or the interval stated by the manufacturer does not yield acceptable results.

However, it is possible to feed the data gathered into a maintenance system and print out schedules electronically.

3.2.3 Schedule Monitoring

Level of Automation: *Full*

It is possible to fully automate the monitoring of maintenance schedules. It does not require human effort. It is possible for the software to prompt the maintenance administrator of an impending planned maintenance service.

3.2.4 Staff Assignment

Level of Automation: *Partial*

It is possible for software to assign staff to maintenance tasks. However, it requires a human being to judge the quality of work from staff to better make an assignment. The availability of the staff is best determined by a human. Based on this, the assignment of staff cannot be fully automated.

3.2.5 Staff Dispatch

Level of Automation: *Full*

Once a member of staff is assigned, software can be used to print the dispatch form with the details of the work to be done. This process can be fully automated.

3.2.6 Maintenance Task Execution

Level of Automation: *None*

The physical execution of a maintenance task cannot be performed by software. Human presence is also required for the inspection of the completed task. This process can, therefore, not be automated.

3.2.7 Reporting Back

Level of Automation: Partial

It is required of a supervisor to check on work that has been completed. The supervisor is also required to sign the dispatch form. The two activities cannot be automated. However, updating the maintenance status of the asset can be automated.

3.2.8 Unplanned Maintenance Requests

Level of Automation: *Partial*

It is possible to eliminate the manual transfer of unplanned maintenance request into an electronic form but it requires human presence to spot the fault and report it.

4 ARCHITECTURE

The GT Maintenance Management System (GTMMS) is the name of proposed system being developed. From the perspective of the end-user, it consists of three major components:

- The Unplanned Maintenance Client
- The Planned Maintenance Unit
- The Maintenance Manager

The Maintenance Manager manages the activities of the Planned Maintenance Unit and the Unplanned Maintenance Client. Figure 4.1 shows an overview of the GTMMS.

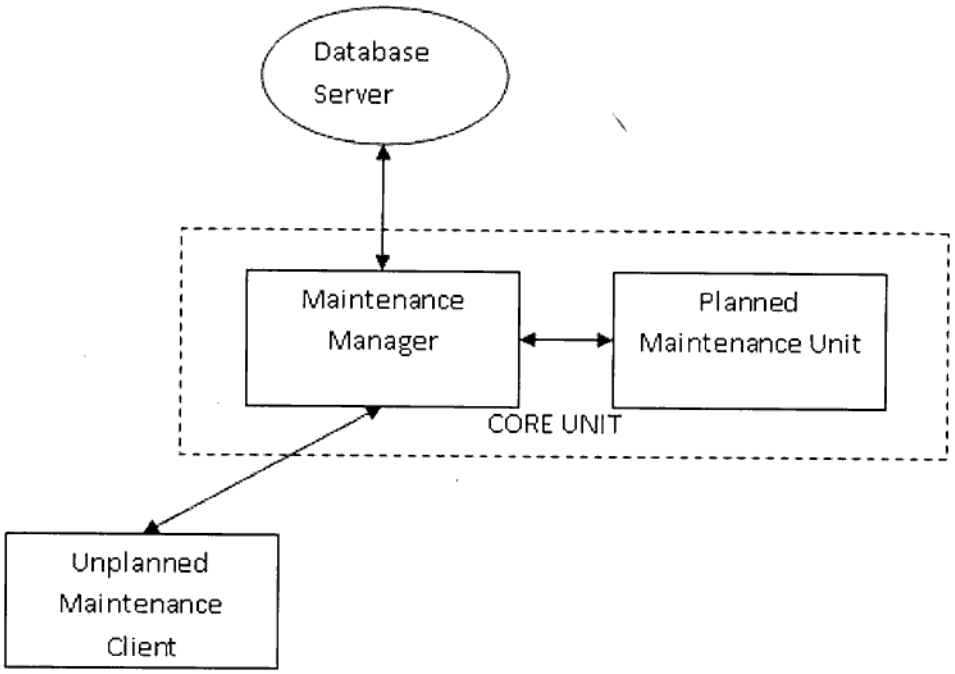


Figure 4.1 GTMMS Overview

The GT Maintenance Management System (GTMMS) has been developed on Client-Server paradigm. It is designed such that each of its segments can be run on different computers across a computer network using the TCP/IP Protocol.

It is segmented into three units:

- Unplanned Maintenance Client (User Requests and correspondence)
- The CORE Unit (Maintenance Manager, Planned Maintenance Unit)
- Database Server

All Cataloguing, Request and Scheduling data are first stored on the Database Server. The Maintenance Manager then retrieves them for analysis. The maintenance administrator/engineer then acts on the result of the analysis (Figure 4.2).

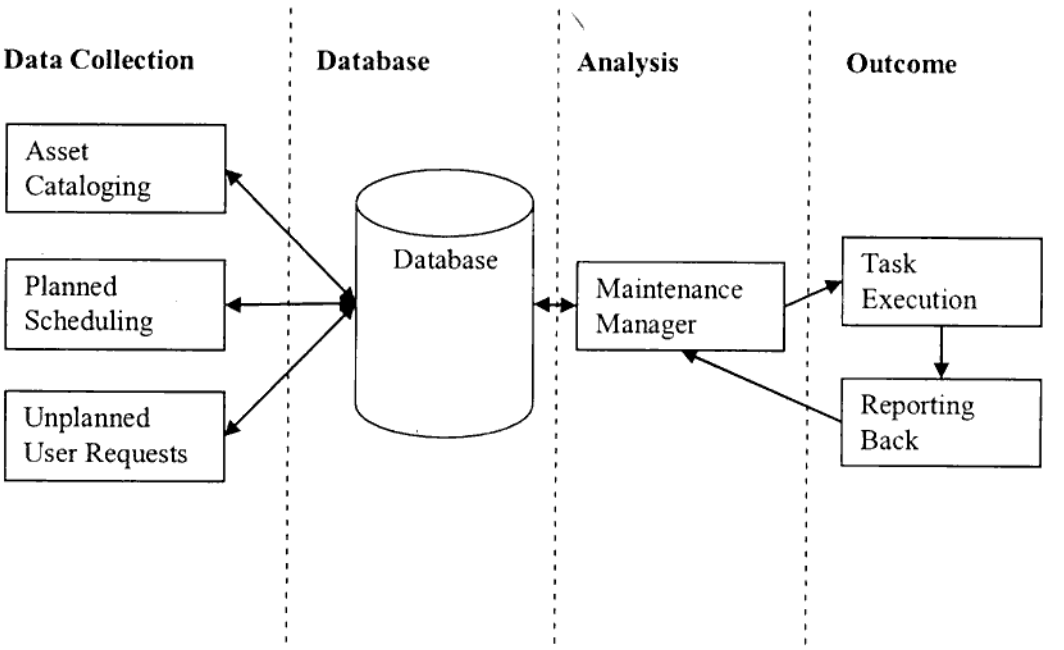


Figure 4.2 GTMMS Architecture

4.1 Unplanned Maintenance Client

The Unplanned Maintenance Client is responsible for taking user requests and sending them to the Maintenance Manager. It consists of a data table (figure 4.3) that houses the users' or departments' login identification details stored on the database server. There is also a data table (Figure 4.4) that stores the correspondences between the user and the maintenance administrator/engineer.

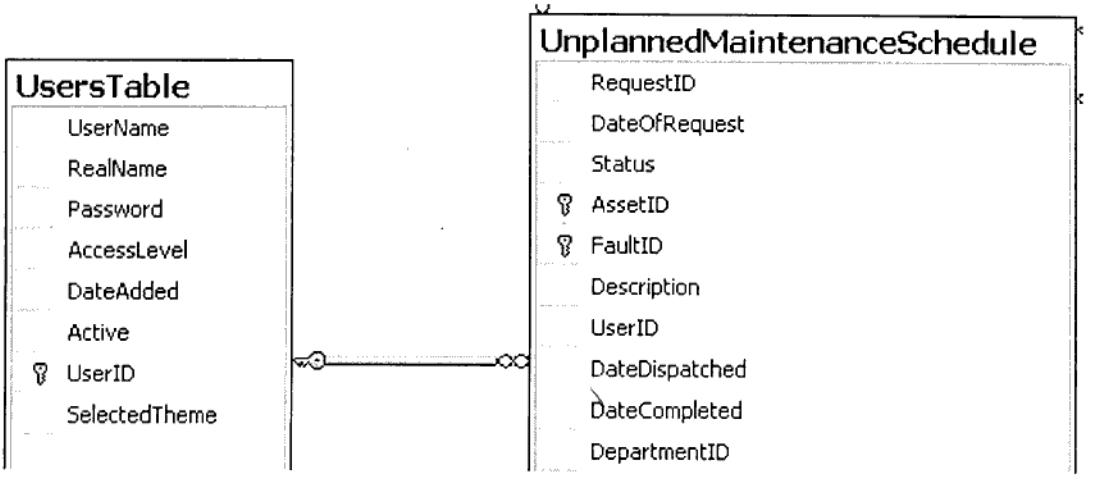


Figure 4.3 User Request Table

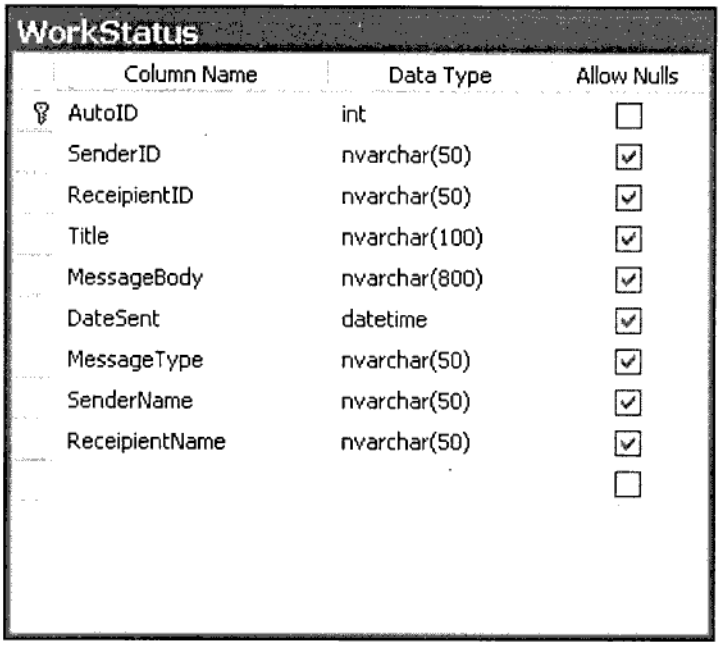


Figure 4.4 User Correspondence Table

Figure 4.5 is the activity diagram of unplanned maintenance.The user fills an electronic requisition form. The form has fields from which the user can select the location, the Asset ID number, the type of fault as well as give a brief description of the problem. The form then goes to the UnplannedMaintenance table. Upon receiving the request, the Core Unit's Maintenance Manager automatically notifies the user of the receipt. The Core Unit's Maintenance Manager then presents the request to the maintenance administrator. The Maintenance Manager also suggests a list of staff with the competence to handle the problem. The maintenance administrator then reviews the suggestions and makes the necessary changes and dispatches the staff to handle the job. The user is also informed of the dispatch via the Maintenance Manager.

Upon execution of the task, the maintenance administrator, after inspection, updates the job status. A report is sent to the user on the status.

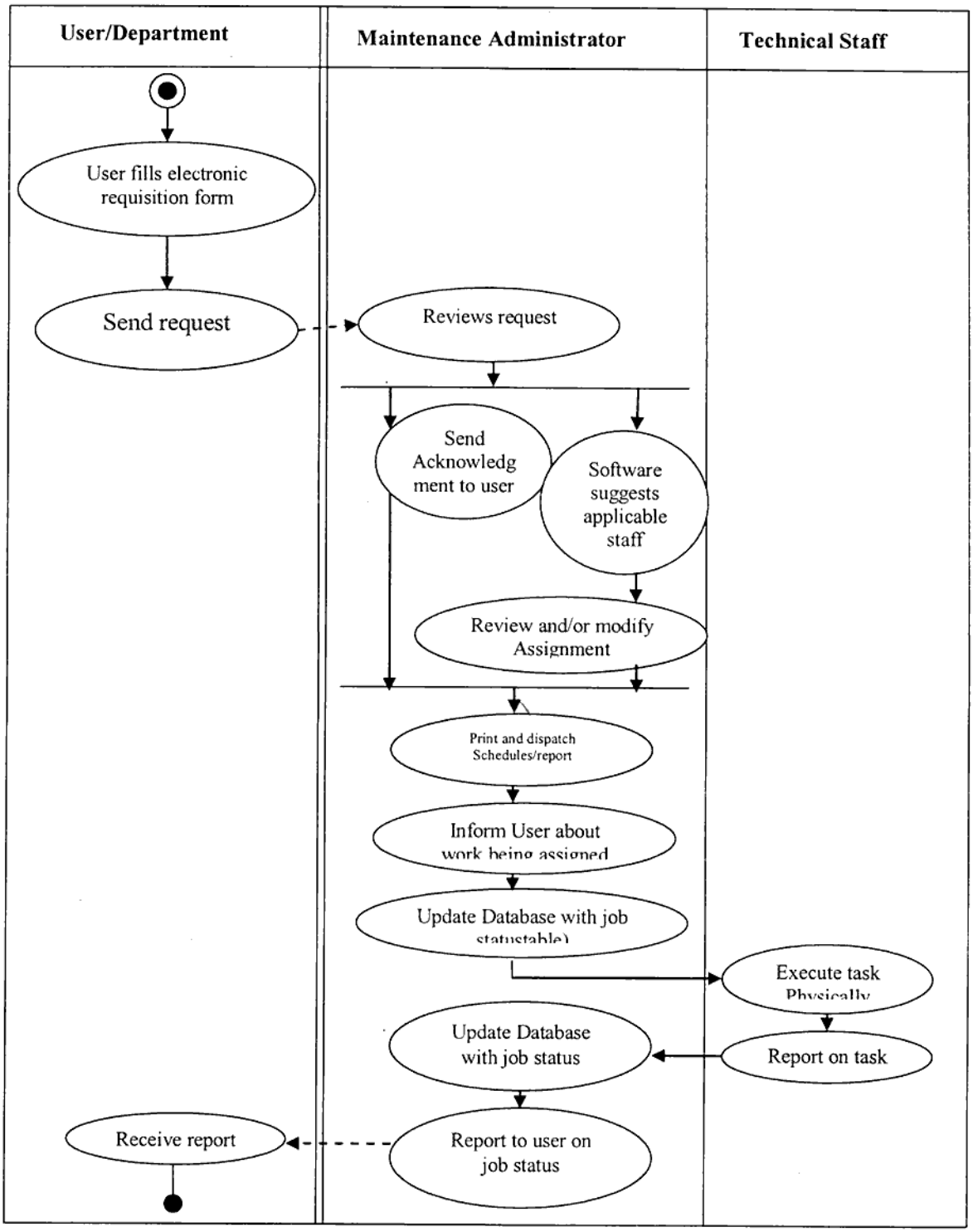


Figure 4.5 Activity Diagram of the Unplanned Maintenance Client

4.2 The Core Unit

The Core Unit is the engine of GTMMS. It is responsible for processing all the maintenance requests as well as scheduling maintenance activities.

The Core Unit has two sections based on their functions. These are:

- The Planned Maintenance Unit
- The Maintenance Manager

4.2.1 Planned Maintenance Unit

The Planned Maintenance Unit is responsible for drafting planned maintenance schedules of assets and monitoring those schedules. It provides an interface for assigning maintenance schedules to assets either individually or as applied to a whole class (group) of assets. The Planned Maintenance Unit integrates seamlessly with the Maintenance Manager for the execution of planned maintenance activities. Figures 4.6 and 4.7 shows the flow and activity diagrams of the Planned Maintenance Unit (PMU)

The following describe the operations of the Planned Maintenance Unit :

- The planned maintenance schedule of the asset is drafted via the PMU
- The PMU reads the schedule and calculates the due date for maintenance
- When a schedule is due, the Maintenance Manager suggests a list of staff with the capability of performing the task to the maintenance administrator.
- The maintenance administrator then selects the staff and assigns them the task. Optionally, the Maintenance Manager could automatically assign a staff based on a fair queue algorithm that the maintenance administrator can review and modify.

- Upon completion of the task, the maintenance administrator updates the status of the schedule.
- The next schedule date is then re-calculated by the PMU

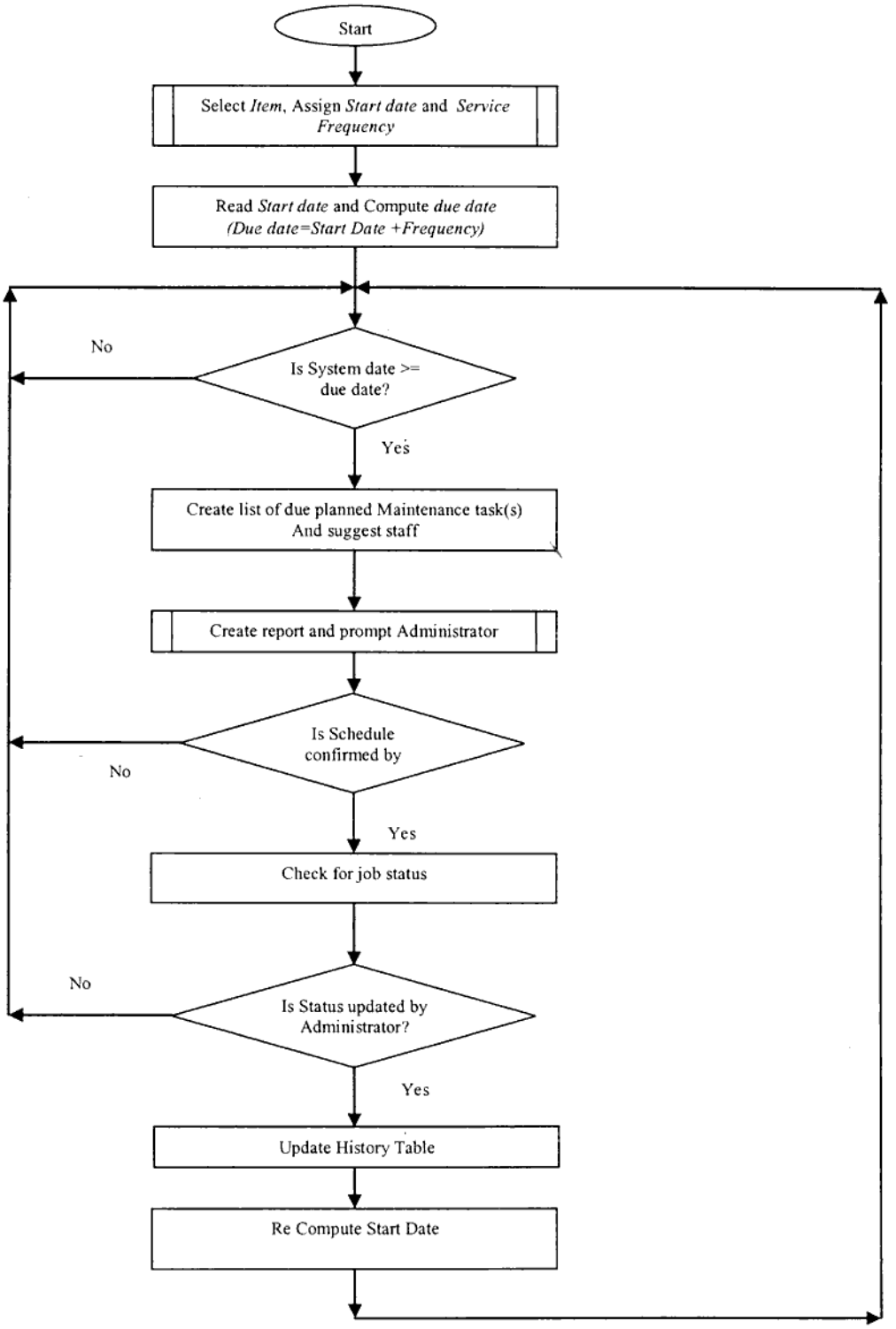


Figure 4.6 Planned Maintenance Flow Chart

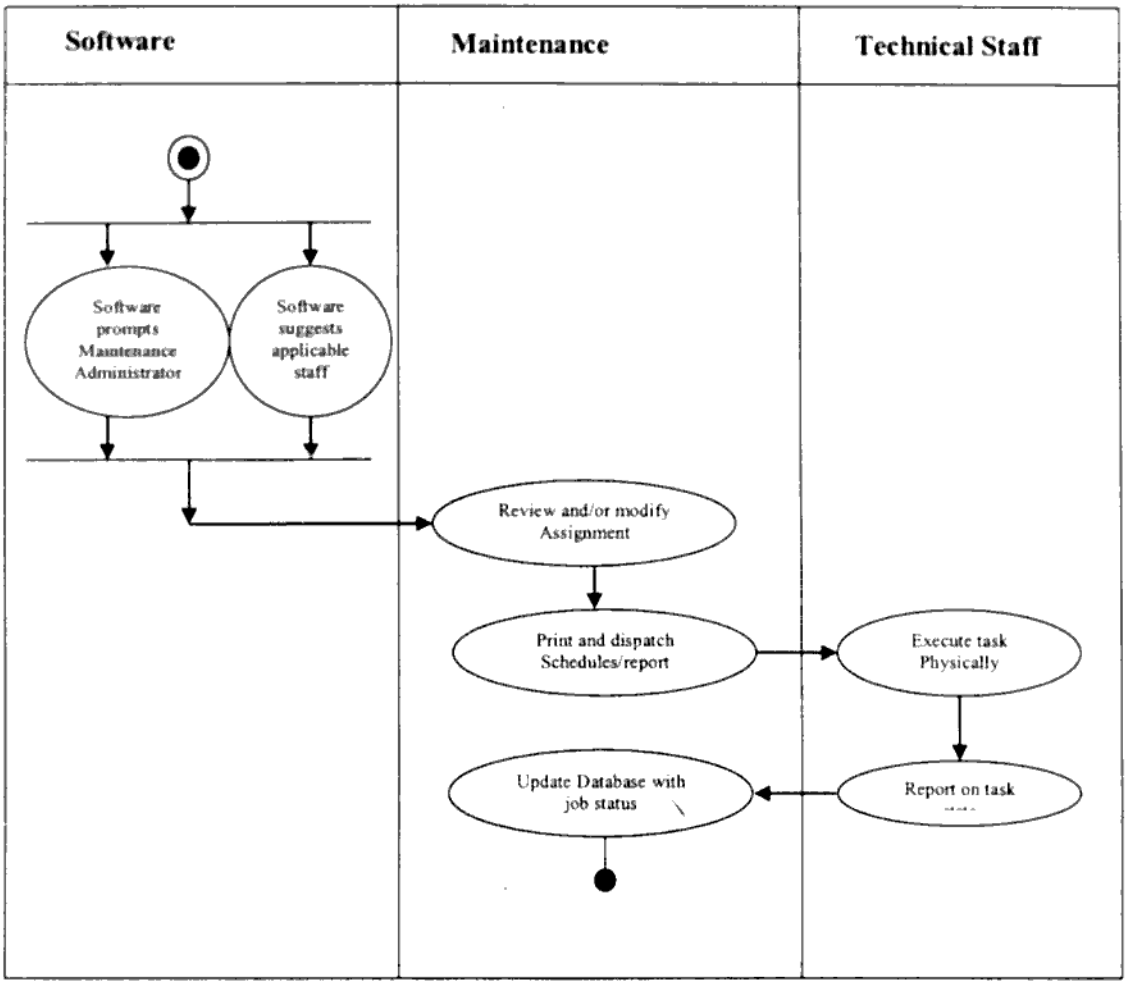


Figure 4.7 Planned Maintenance Activity Diagram

4.2.2 The Maintenance Manager

The Maintenance Manager (MM) is the bedrock of the GTMMS. It coordinates integrates all the other components of the software. It is responsible for the following:

- Cataloguing of Assets:

  The Maintenance Manager is responsible for classifying assets into segments such as, categories, types, classes and brands.

- Generating Room Numbers

  The MM implements a function for generating room numbers for the storage of assets. There is also a module for locating rooms.

- Managing Staff Work Shifts

  The Maintenance Manager implements a time table module that enables the maintenance administrator manage the work shifts of the technical staff.

- Managing Staff Competencies (skills)

  There is a module for managing the capabilities of technical staff.

- Managing Faults and Services

  The Services and Faults module presents a list of services required for planned maintenance activates as well as a check list of the various step to be taken when servicing.

- Managing User Accounts

  The Maintenance Manger manages the accounts of user. This involves the creation and modification of the account and the assignment of privileges.

The Maintenance Manager is also responsible for tasks such as acknowledging users about the status of their requests, suggesting applicable staff for a task as well as automatically assigning staff.

4.3 Database Server

The hub of all the data involved in the software resides on the database server. The server incorporates:

- Data tables for storing data for later retrieval
- Data Views for looking at the data through certain filters and conditions
- Functions for performing some simple computational tasks
- Login Accounts for restricting access to the data

5 SOFTWARE AND DATABASE COMPONENT SPECIFICATION

5.1 Software Components

After observing and interviewing the staff on their maintenance activities, the following requirements were realised:

- To eliminate the need for manual checks on preventative maintenance schedules,
- To enable departments and individuals forward maintenance requests directly to the system.
- To provide information that can be used for decision making

From the points listed above, the following components (functionalities) were crafted out:

5.1.1 Categorizing assets for easy identification

The asset needs to be placed in various categories and sub categories for differentiation and easy access. Information that uniquely identifies each asset needs to be created. The location of the assets should also be known at every instant.

5.1.2 Cataloguing Staff and their competencies

Information on staff and their areas of expertise is needed for the selection of the appropriate candidate for a maintenance task. Information of their working shifts is also needed for the maintenance administrator to better assign a task.

5.1.3 Generating automatic maintenance schedules

A key requirement to eliminating the need for manual checks on preventative maintenance schedules is to be able to generate an automatic maintenance schedule. This is based on monitoring the preventative maintenance intervals and selecting the assets that are due for servicing.

5.1.4 Managing Staff Information

Information on technical staff is required for the selection of the appropriate staff for a task assignment. This information consists of the Staff general information, time-table, skills and the staff job burden (number of tasks assigned or performed).

5.1.5 Tracking Maintenance History

Tracking of the history of maintenance is essential for determining problematic assets and the cost of maintaining assets. This component is very important in generating reports for decision making.

5.1.6 Receiving requests from remote ends

The software is required to provide the functionality of receiving requests from departments and individuals. This cuts down the need to physically move from one location to the other. This functionality reduces paper work and delays in reporting an incident. The requesting party should also be given progress reports of the status of the request at each stage.

5.1.7 System Actors and their Corresponding Functions

The system actors are the end users of the Maintenance Management System. They have varying amount of privileges and responsibilities assigned to them on the use of the software. Below are the primary users and their responsibilities

Software Administrator(s)

- Installs database management software for data management
- Configures database to communicate with software application
- Installs software application
- Creates user account(s) and sets user privilege(s)
- Performs update or delete or add functions. (-basically, all tables)
- Views reports and creates new reports/user defined queries
- creates updates when necessary
- controls user passwords and usernames

Maintenance Administrator(s)

- Creates user account(s) and sets user privilege(s)
- Performs Update or Delete or Add functions
- Views reports
- Assigns staff to job(s)
- Assigns interval(s) of frequency (ies) to job(s)
- Prints reports

- Update system with job states/status
- Controls user passwords and usernames

Simple users

- View(s) reports
- Send(s) request(s)
- Sends/receives acknowledgments

Technical Staff

- Executes task(s) physically and report on job status/state

5.2 Database Specification

The database is the backend of the system that stores all data used by the system. Data from users are stored within the database. This data is then processed into a meaningful form that can be used by the user for further action.

The following are the designs of the core database tables.

Table 5.1 shows a summarized view of the kind of data required for categorization.

Classification Scheme	**Members**	**Description**
Category	Category ID Category Name Category Description	This creates the major categories for classifying assets (e.g. Vehicles, HVAC)
Type	Type ID Type Name Type Description	This further breaks downs categories into sub categories (e.g. SUV, FloorStanding)
Brand (e.g. Toyota)	Brand ID Brand Name Brand Description	This segments assets by manufacture (e.g. Toyota, LG)
ItemClass(Group)	ItemClassCode Type ID Brand ID	This groups the Category, Type and Brand together to form a Class of an asset
Location	Location Code Block ID Floor Room Number	This stores the location of each asset
ItemSpecification	Asset ID ItemClassCode Location Code	This stores information about a specific asset. It is uniquely identified by its Asset ID

Table 5.1 Summarized view of data required for categorizing assets

Table 5.2 shows a summarized view of the tables that are used in maintaining staff information and assignment. Assignment of staff is based on the skills of the staff, and the work-shift of the staff.

Classification Scheme	**Members**	**Description**
Staff	Staff ID Surname OtherNames Address	This gives a general information about staff
StaffSkill	StaffSkillID Staff ID SkillID	This maps a member of staff's unto his competencies (skills)
Skill	Skill ID Skill Name Skill Description	This contains a list of competencies that a member of staff may possess or is required for servicing an asset
StaffShift	Staff ID Shift Code	This maps staff unto their work shifts
Shift	Shift Code Begin Time End Time	Stores a list of available work-shifts

Table 5.2 Summarized view of data required for staff assignment

Table 5.3 lists the tables that are used in generating planned maintenance schedules. The Maintenance Schedule is linked to the Service by the Service ID. The Service Work-Plan is also linked to the type of service by the Service ID. The work-plan lists the steps required for servicing an asset

Classification Scheme	Members	Description
Maintenance Schedule(Planned)	Schedule ID Asset ID Service ID Start Date Frequency Next Due Date ...	This contains scheduling information for each asset
Service	Service ID Service Name Service Description	This contains a list of services required for preventative maintenance
Service WorkPlan	Service ID Service Step Number Service Task Remarks	This lists the steps (checklist) for performing a service

Table 5.3 Summarized view of data required for automatic scheduling

In Table 5.4, the history of assets that have undergone maintenance are stored in the History Table. Staff that performed those maintenance are listed in the HistoryStaff table. The History Table and the HistoryStaff table are linked via the HistoryID

Classification Scheme	Members	Description
History Table	History ID Date ItemClassCode AssetID ServiceID DateDue HoursOverDue MaintenaceCost DateOfAssignment ScheduleID RequestID	This stores the maintenance history of all asset that have been serviced and associated cost
HistoryStaff	HistoryID StaffID SkillID	Keeps the history all staff that have performed a maintenance task

Table 5.4 Summarized view of data required for tracking the history of maintenance

For users to send unplanned maintenance request, their credentials are stored in the UsersTable. This facilitates communication between the maintenance administrator and the user. The WorkStauts table keeps stores the messages sent between the user and maintenance administrator or the system. Common faults are retrieved by users for the faults table. And the request is scheduled in the unplanned maiantenace schedule. The SenderID of theWorkStatus links it to the UsersTable's UserID. The unplanned maintenance schedule is linked to the Fault table via the Fault ID. Table 5.5 shows the tables used in this interaction.

Classification Scheme	Members	Description
UsersTable	User ID User Name Password ...	This contains the information about the user or department making the request
WorkStatus	Sender ID Recipient ID Title Message Body Message Type ...	This stores the messages exchanged between the requesting party and the maintenance administrator.
UnplannedMaintenance Schedule	Request ID Date Of Request Status Asset ID Fault ID Description User ID	This contains the list of all the unplanned maintenance requests by the users or departments. The status of the request is also indicated. Type of fault is also indicated by the requesting party
Fault	Fault ID Fault Name Description	This contains a list of common faults that a user or department can select from or a custom fault that a user can specify

Table 5.5 Summarized view of data required for maintenance requests

Figure 5.1 shows the essential tables used by the software and their linkages.

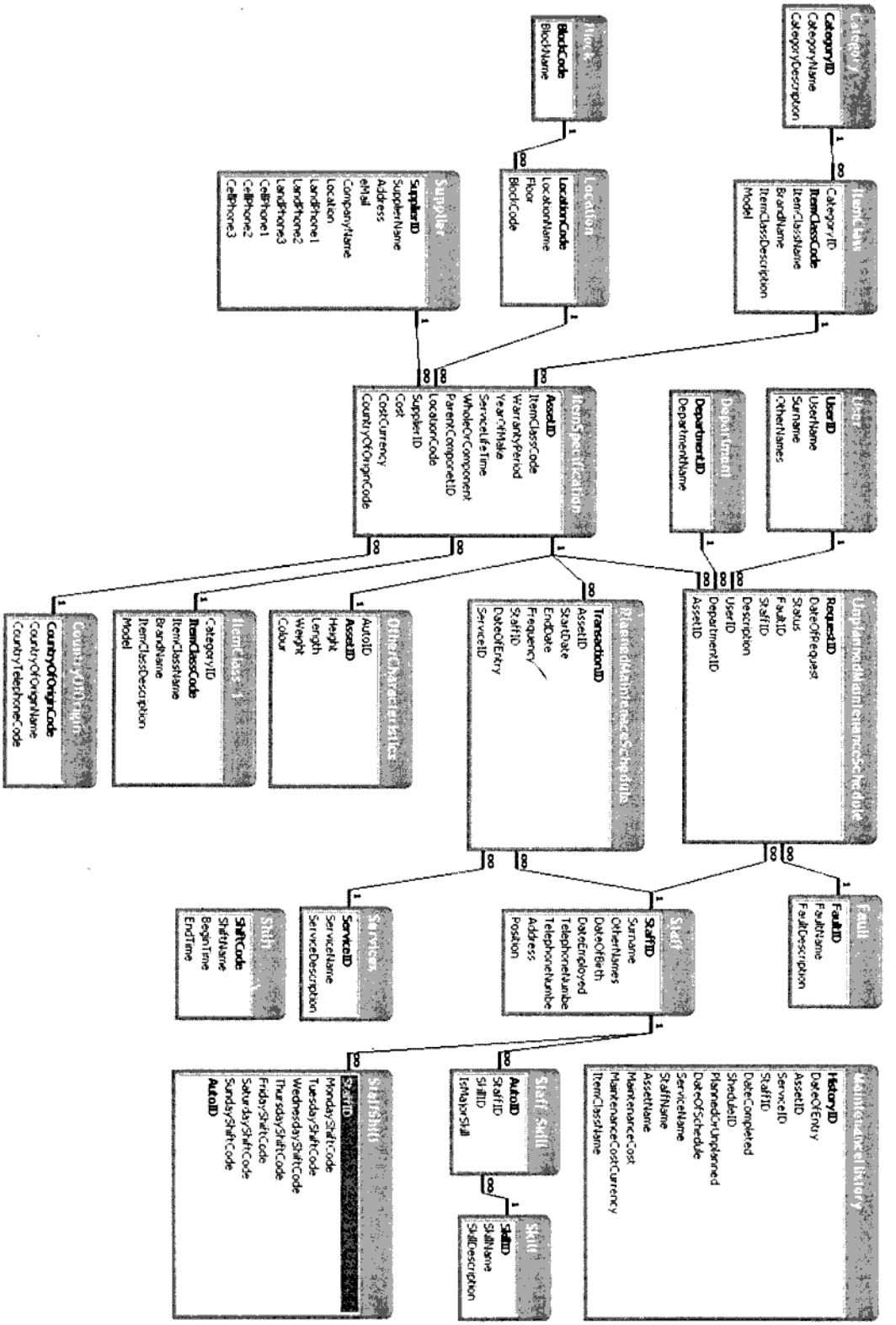


Figure 5.1 Database tables and their linkages (relationships)

6 DESIGN AND IMPLEMENTATION

6.1 Choice of Operating System Platform, Programming Language and Database Sever

Two operating system platforms were considered .The first is the Linux operating system and the second Microsoft Windows. Microsoft Windows operating system was chosen due to its wide user base and the relatively higher availability of software development tools for its environment. Linux is much more complex to administer and is not as widely used as Microsoft Windows.

6.2 Choice of Database Server

Three database servers were considered. These are Microsoft Access, Oracle and SQL Server 2005.

Microsoft Access was not chosen because it cannot handle a large number of connections and does not scale very well as the database grows.

Oracle requires quite a considerable amount of technical knowledge before use. Sybase is not very user friendly.

Microsoft SQL Server 2005 Express edition was chosen because it is stable, easy to install and administer and it is available for free and it is easier to use. It also does not require a high learning curve. It was also chosen because it is scalable and robust.

6.3 Choice of Programming Language

Microsoft Visual Basic 2005 (Express Edition) was the choice of programming language for development. It is an object oriented programming language and well suited for rapid application development. It is a powerful tool for developing Graphical User Interface applications (GUI). It is also an event driven programming language. A method can be coded to handle the events. Visual Basic 2005 (Express Edition) is relatively easy to use, stable and user-friendly. It is also available for free.

Krypton Toolkit, an extension by Component Factory written for the Microsoft Visual Basic environment, was used in many of the user interface designs. It provides a functional user interface with the support for multiple themes.

6.4 GT Maintenance Management Software

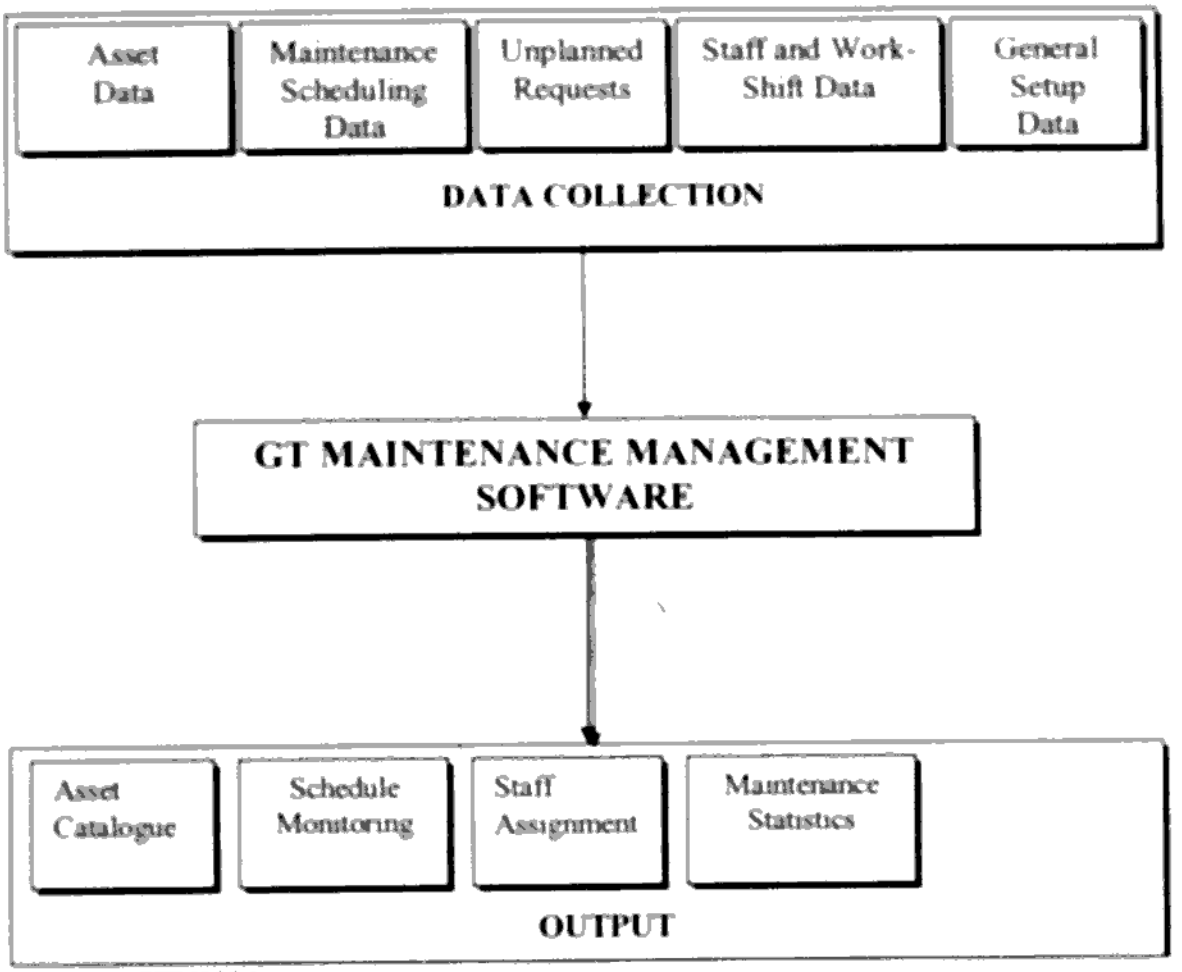


Figure 6.1 GTMMS Layout

The Layout of the GTMMS is shown in Figure 6 4 based on the software architecture discussed in Chapter 4.

The Data Collection block deals with the collection of data for cataloguing, scheduling, staff work-shifts and setting up of the software.

The GTMMS works on the collected data to produce an asset catalogue monitor maintenance schedules, suggesting and assigning staff as well as providing maintenance statistics and inventory information.

6.4.1 Organisation

The GTMMS is divided into two physically separate applications.

- The Unplanned Maintenance Client
- The Core Unit

The Core Unit is further partitioned into five major blocks or modules according to their functionalities. These are:

- Item Classification
- Item Specification
- Maintenance
- Staff
- Setup

The Unplanned Maintenance Client

This is primarily used for collecting unplanned maintenance request. The process is initiated by a user logging in with his credentials (user-name and password) as shown in Figure 6.2.

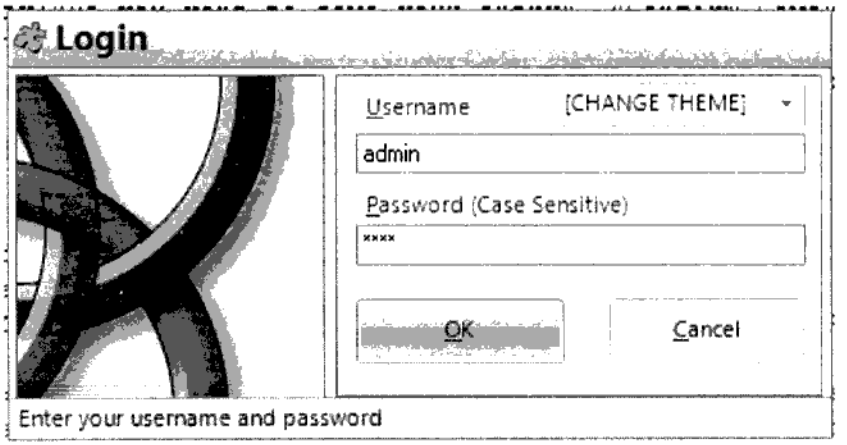


Figure 6.2 Login Screen

The user is then logged onto the Unplanned Maintenance Client. A list of previous correspondences, if any, is displayed on an email-like form as shown in Figure 6.3.

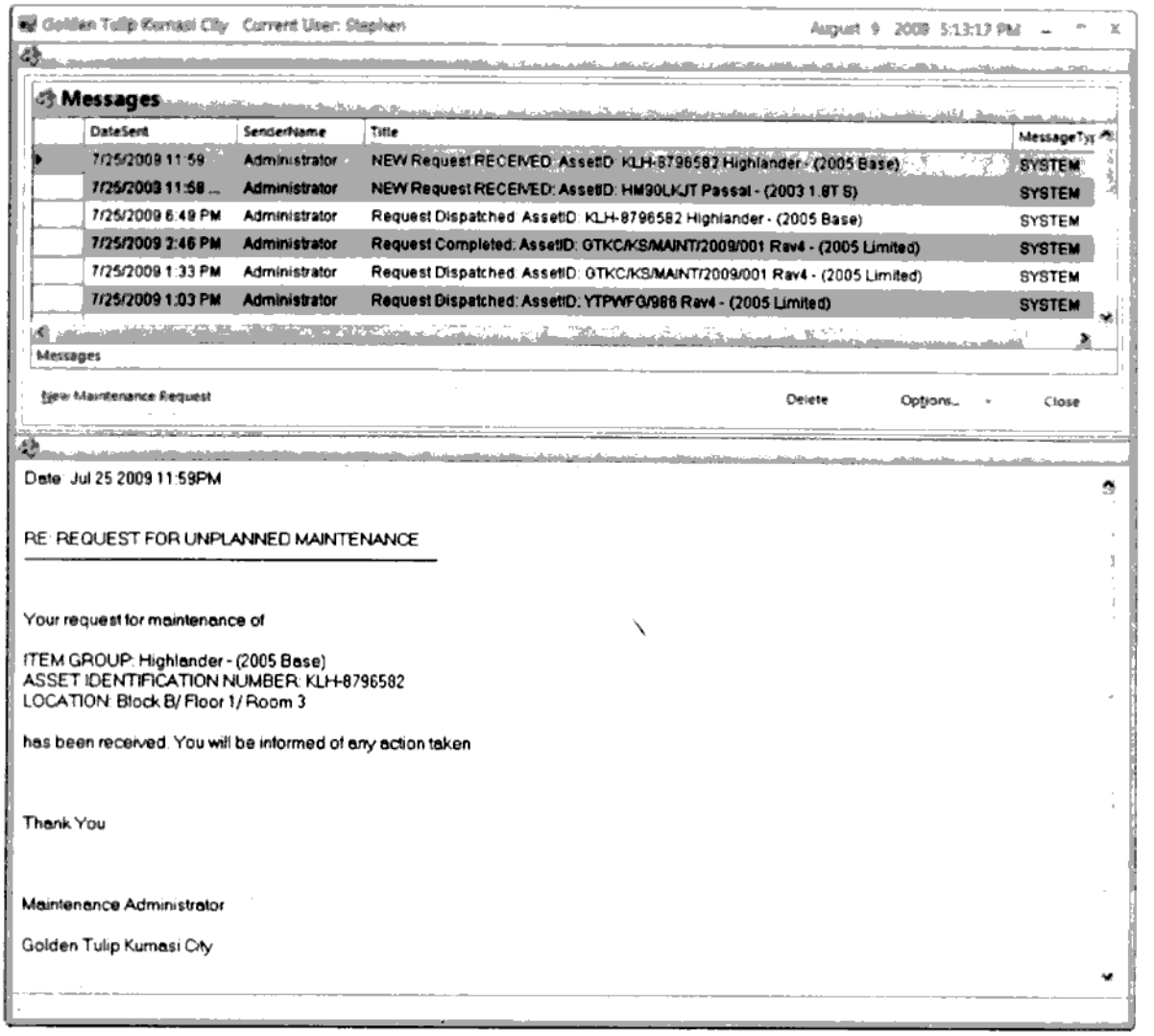


Figure 6.3 Unplanned Maintenance Client

To place a new request, the user first locates (Figure 6.4) the room and selects the item (asset) about which he wishes to report the defect. Selecting a location automatically displays all the (catalogued) assets stored in that location.

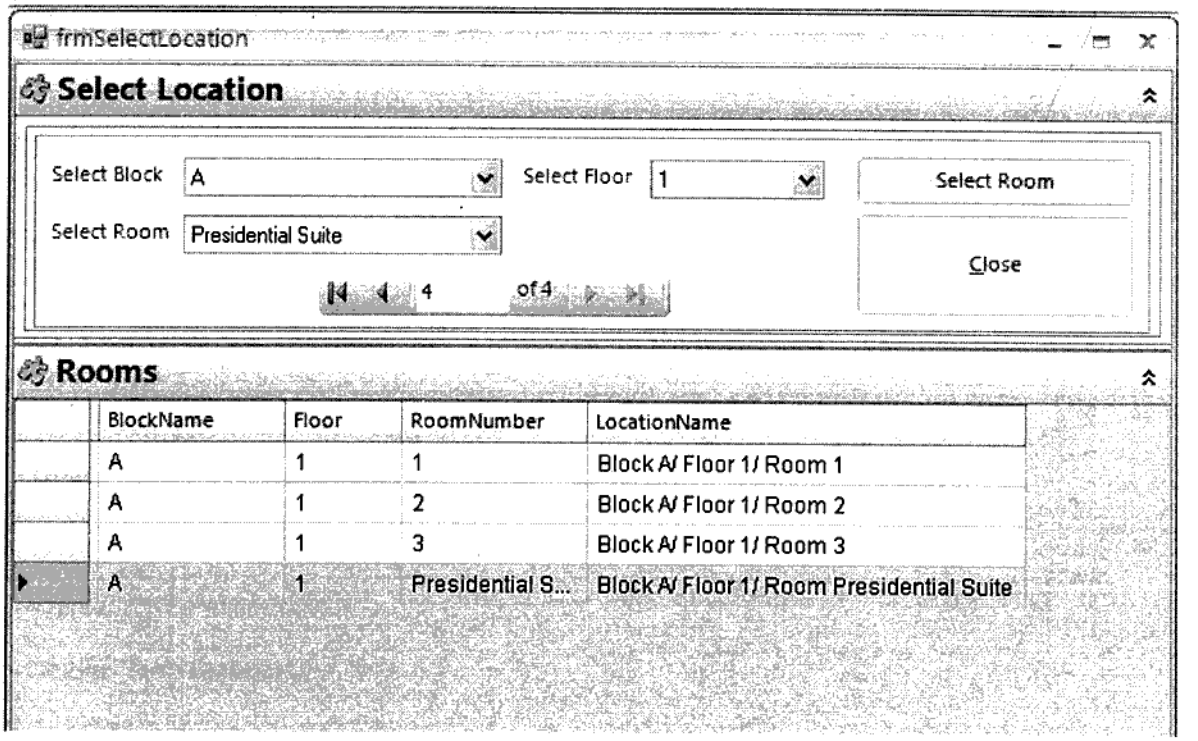


Figure 6.4 GTMMS Room Locator

The user can then select the type of fault and optionally, add a description of the problem for submission (Figure 6.5). The submitted request is then acknowledged automatically of the receipt (Figure 6.6)

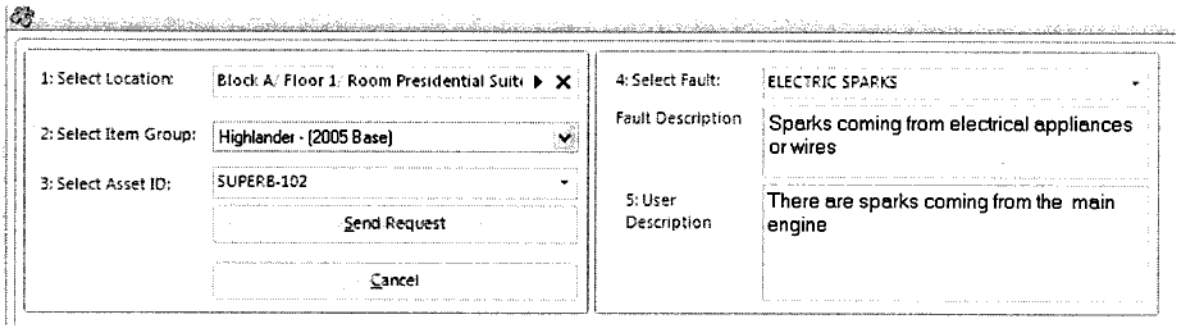
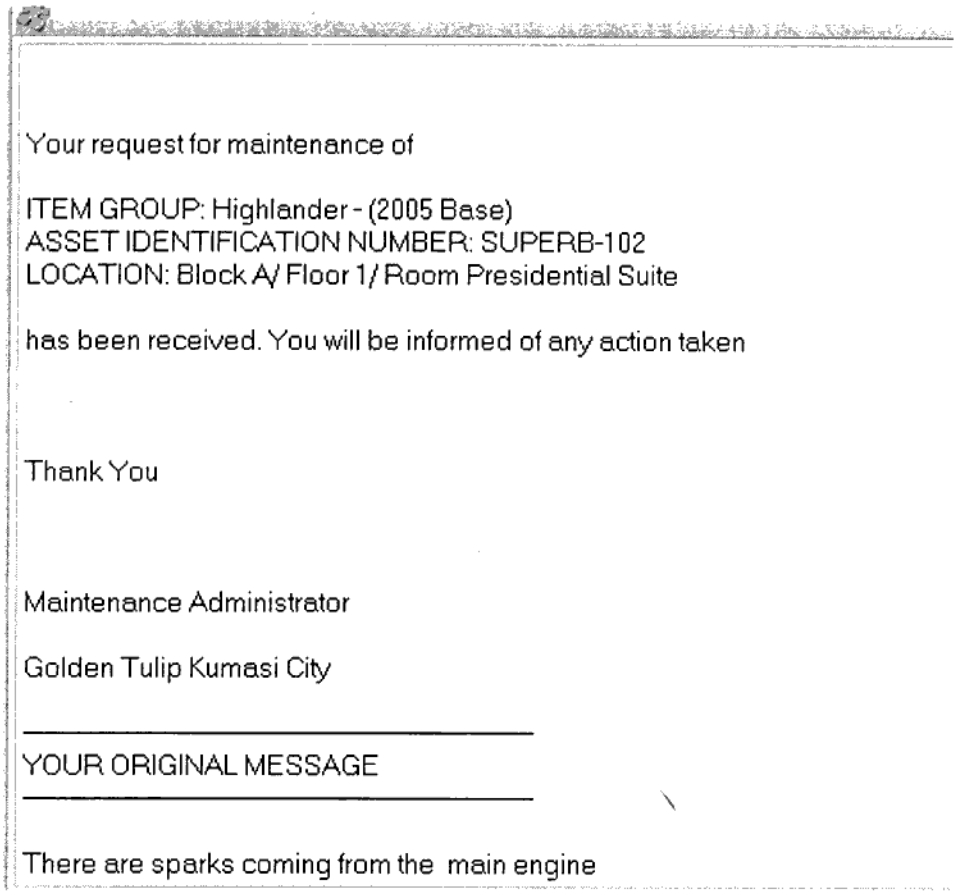


Figure 6.5 Fault submission


Your request for maintenance of

ITEM GROUP: Highlander - (2005 Base)
ASSET IDENTIFICATION NUMBER: SUPERB-102
LOCATION: Block A/ Floor 1/ Room Presidential Suite

has been received. You will be informed of any action taken

Thank You

Maintenance Administrator

Golden Tulip Kumasi City

YOUR ORIGINAL MESSAGE

There are sparks coming from the main engine

Figure 6.6 Acknowledgement of unplanned maintenance request

The rest of the unplanned maintenance processing is taken up by the Maintenance Manager.

The Core Unit

The Setup block

The Setup block provides the functionality of setting up the GTMMS for use. It comprises the Room Number Generator for generating room numbers, Services and Faults setup for predefining a list of services for scheduling maintenance activities, Service Work-Plan for setting up the checklist for a maintenance task, and a User

Manager for managing the user login and access rights to the system. It also has a Supplier, Country of origin for storing supplier and country information.

Room Number Generator: The Room Number Generator works by allocating room numbers on a floor by floor basis. The room numbers may be sequential over a range of numbers, or can be named as shown in Figure 6.7.

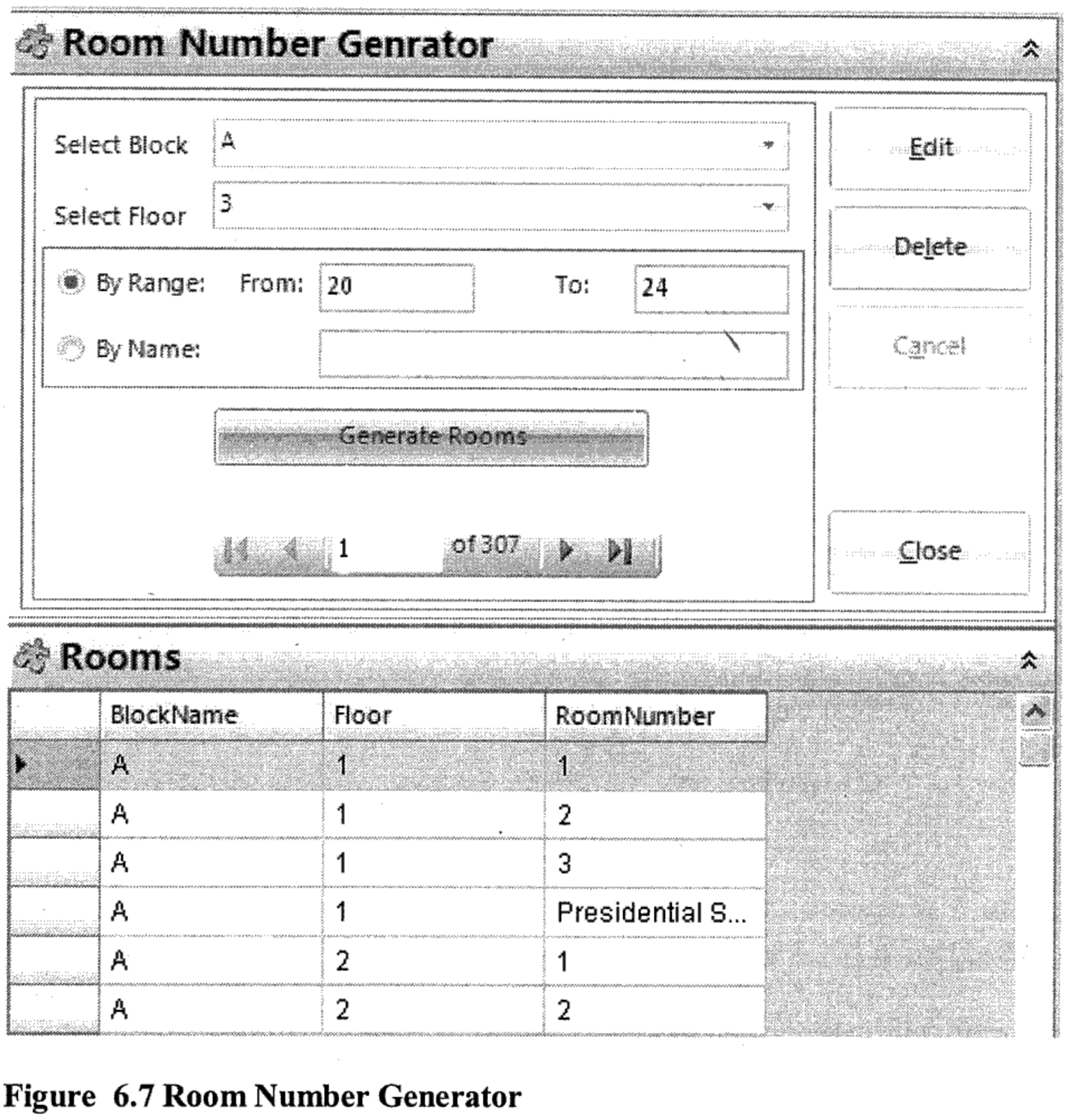


Figure 6.7 Room Number Generator

Services and Faults: These two modules are technically the same but functionally different. The Services module is for managing all services that are

Manager for managing the user login and access rights to the system. It also has a Supplier, Country of origin for storing supplier and country information.

Room Number Generator: The Room Number Generator works by allocating room numbers on a floor by floor basis. The room numbers may be sequential over a range of numbers, or can be named as shown in Figure 6.7.

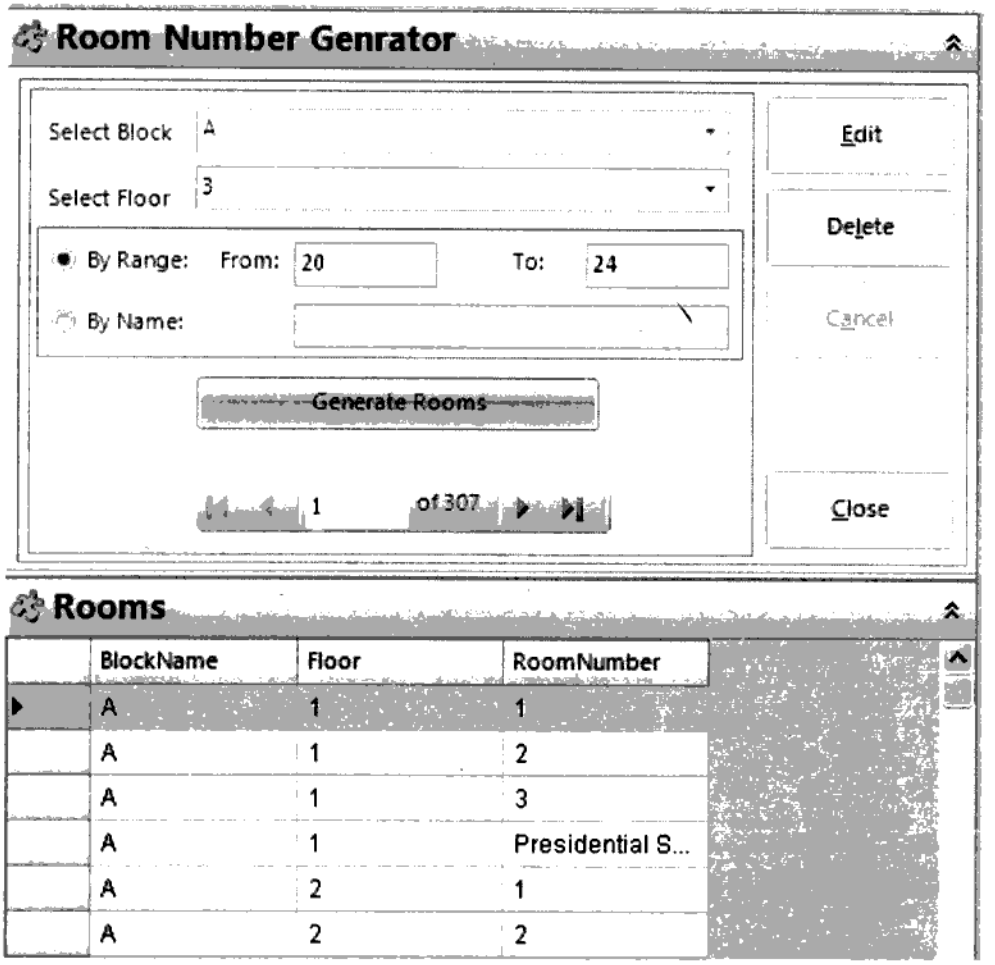


Figure 6.7 Room Number Generator

Services and Faults: These two modules are technically the same but functionally different. The Services module is for managing all services that are

necessary for planned maintenance scheduling, The Faults module is used for managing common faults that may arise from unplanned maintenance requests *(See Figure 6.8)*.

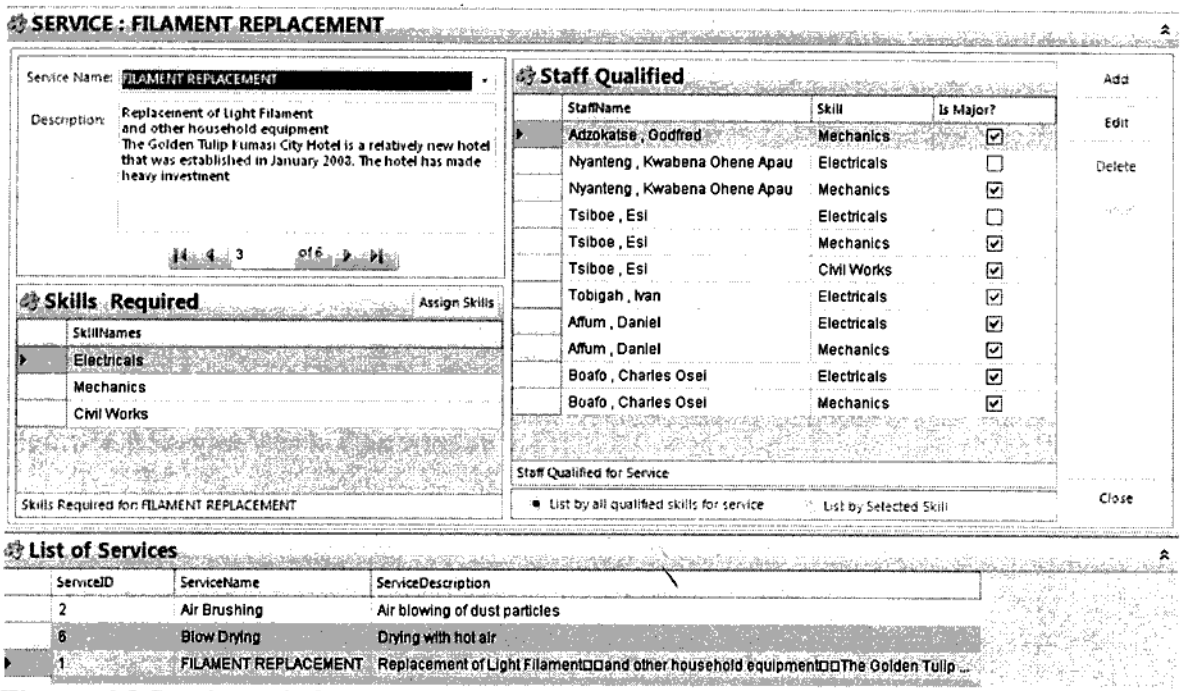


Figure 6.8 Services window

Service Work-Plan: This module manages the list of the various steps that are necessary in carrying about a service *(See Figure 6.9)*

User Manager: The User Manager (See Figure 6.10) manages the addition, modification or removal of user accounts and access rights into the GTMMS. All the passwords are stored in the database using a hash function for security

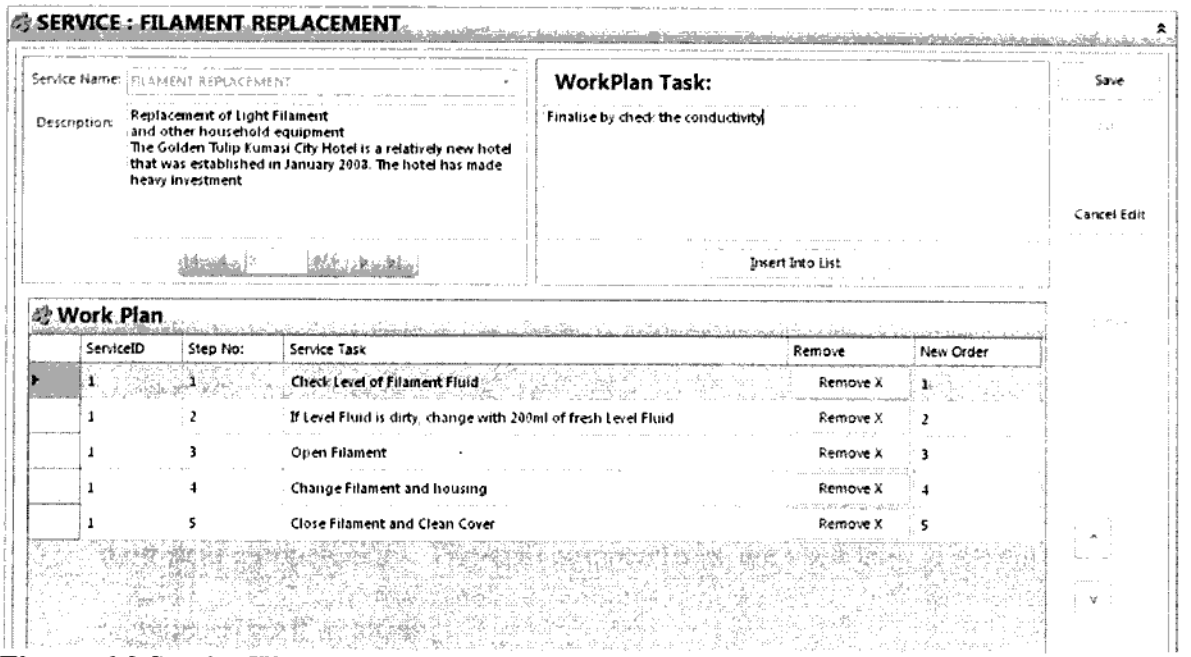


Figure 6.9 Service Work-Plan window

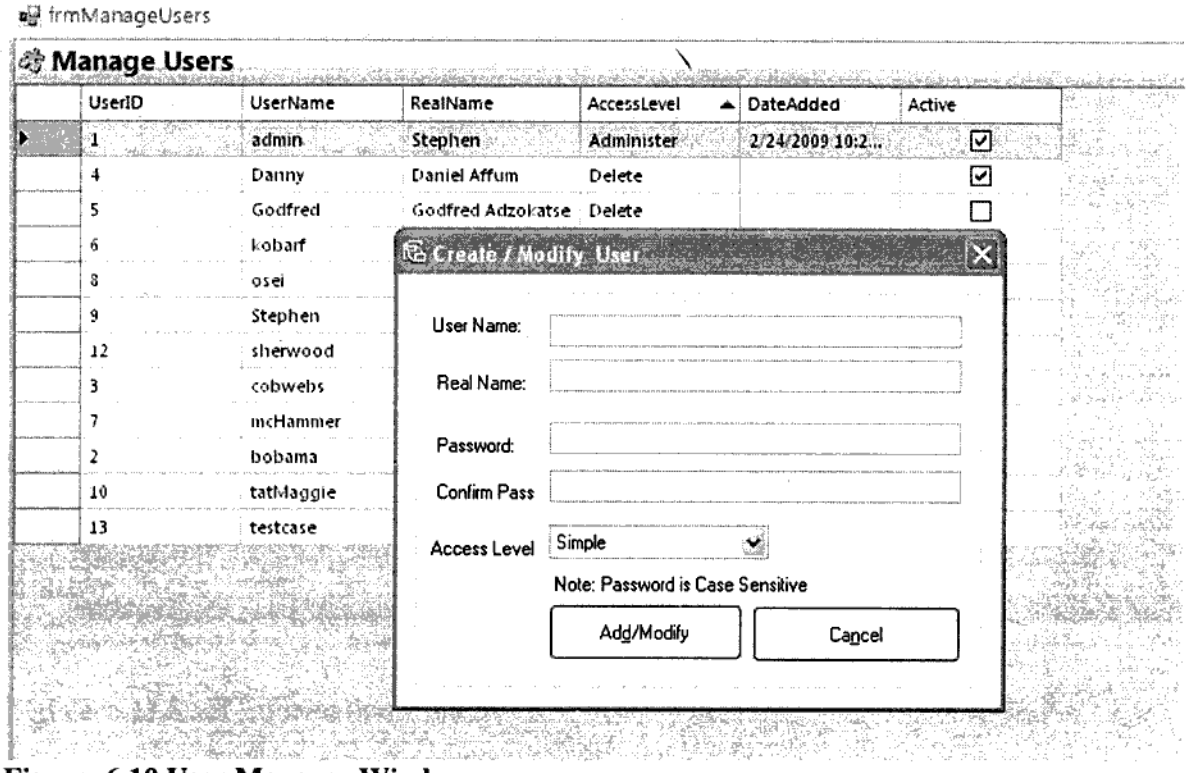


Figure 6.10 User Manager Window

Staff Block

The Staff block provides the functionality of managing the technical staff responsible for carrying out the physical maintenance tasks. It comprises the Staff information module, the Skills module, and the Work Shift module.

Staff Information: This module (Figure 6.11) is used for storing staff details such as names, date of birth and contact addresses etc. It is also used for assigning their skill competencies. The skill competencies are used later when assigning a member of staff to a task.

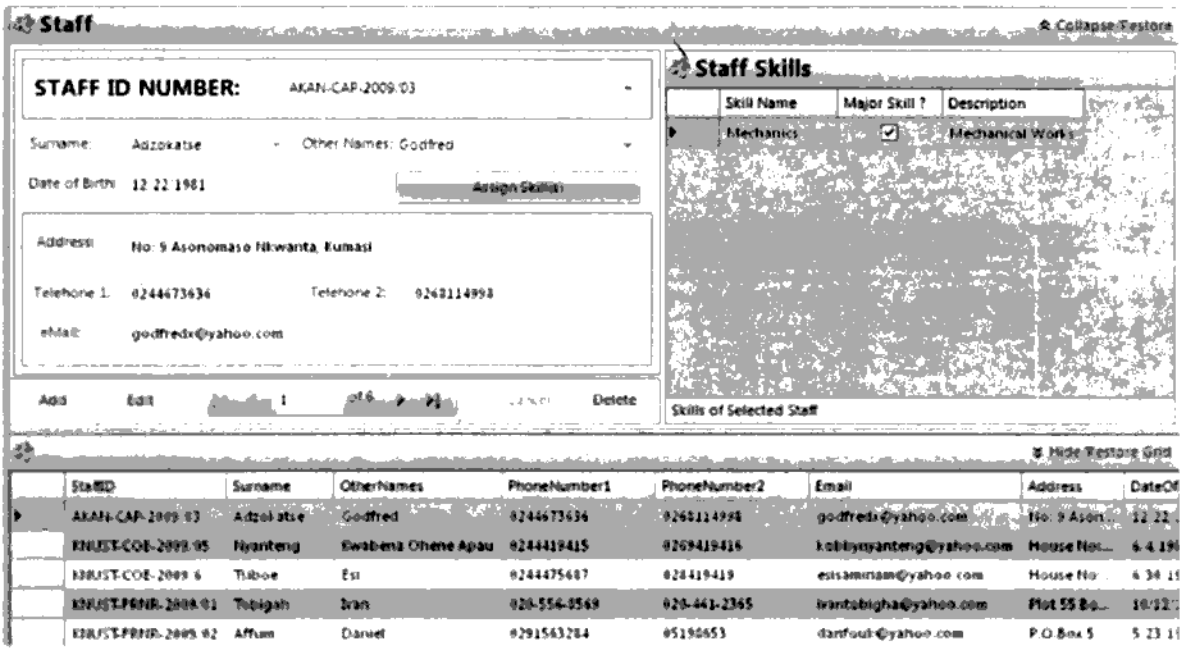


Figure 6.11 Staff Information window

Skills: The Skills (Figure 6.13) module manages the list of skills required to perform a maintenance task. The skills are then assigned to staff with that qualification.

Work Shift: This module (Figure 6.12) manages the working hours of staff. It provides the maintenance administrator the convenience being able to tell when a member of staff is available for work when scheduling.

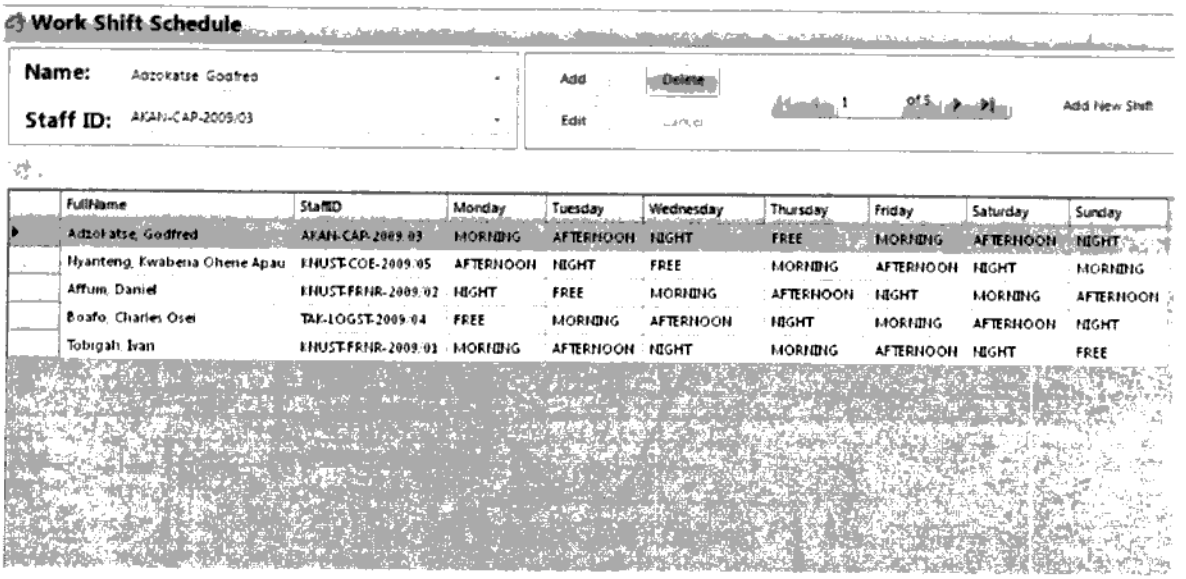


Figure 6.12 Work Shift Schedule

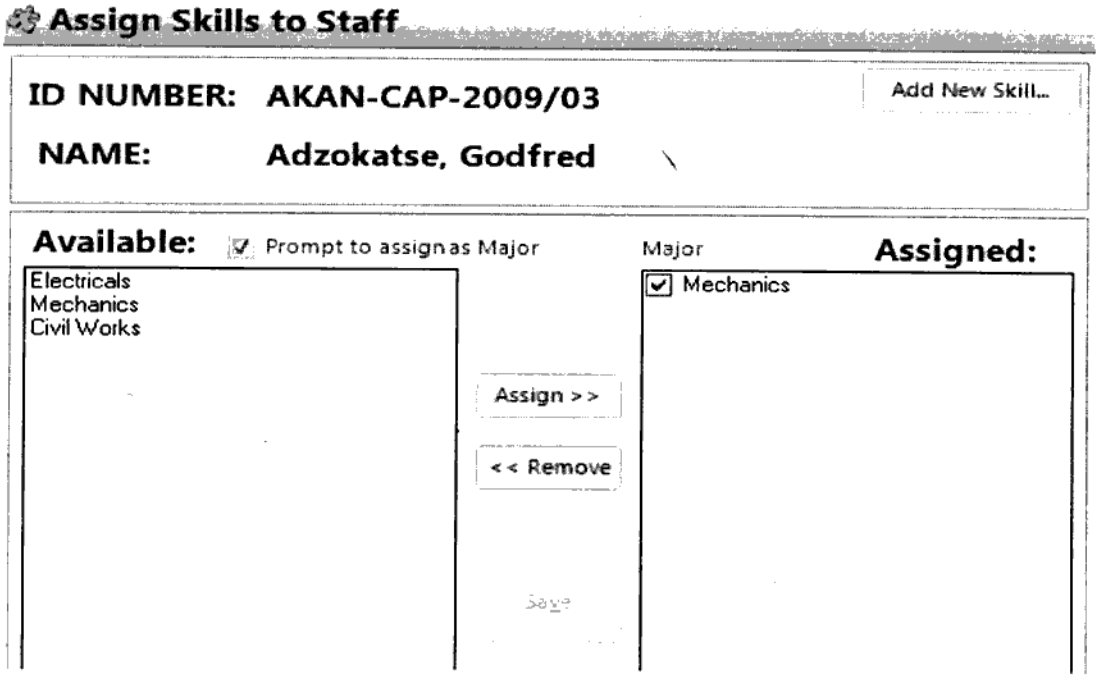


Figure 6.13 Skill Assignment window

Item Classification Block

The Item Classification block is used for setting up a categorization scheme of assets. This block does not create an inventory of assets and is solely concerned with classification. It is divided into Category, Type, Brand and Class (Group) categories (Figure 6.14).

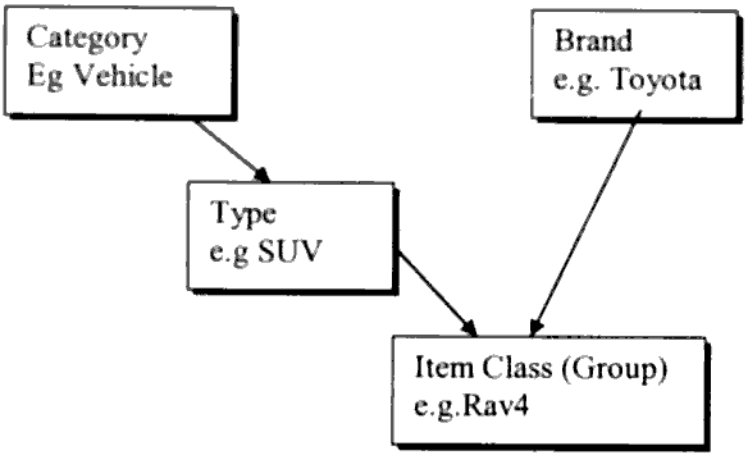


Figure 6.14 Asset Classification Scheme

Category: The Category module (Figure 6.15) is a larger grouping of assets. Assets belonging to different categories have very little in common if any at all. For example, the category "Vehicle" has little in common with the category "Furniture".

Type: The Type module (Figure 6.16) narrows down the category to those with some major characteristic in common. Items classified at the type level may have identical functionalities but different implementation or appearance. For

example, an Sports Utility Vehicle (SUV) and a Saloon car belong to the same Category (Vehicles) with identical functionalities but implemented differently.

Brand: This module groups 'Types' according to manufacturers. For example, Toyota, Ford, VW may produce items belonging to the same 'Category' and 'Type'. The Brand is independent of the 'Category' or 'Type' since one manufacturer may be producing several items in many Categories.

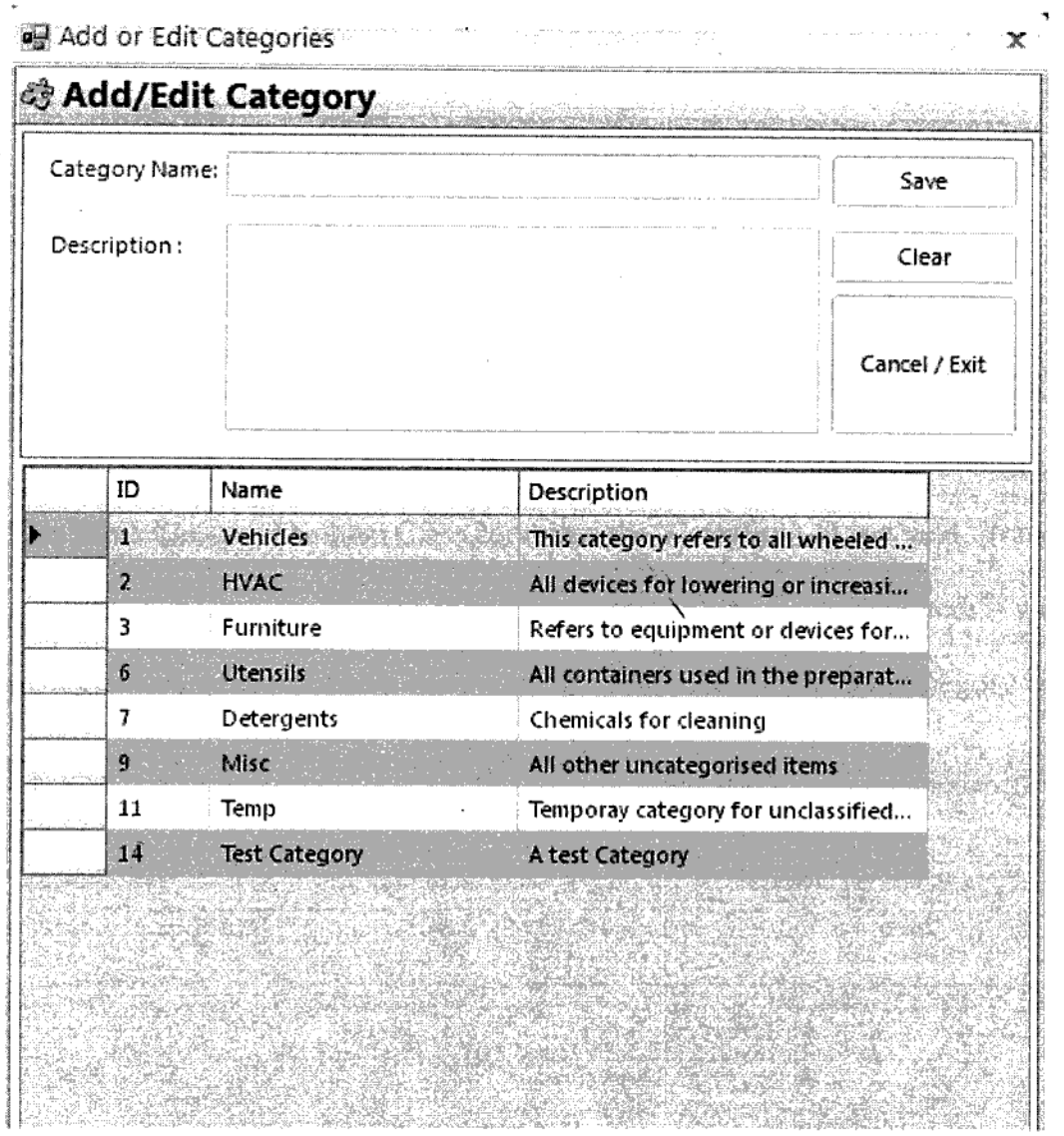


Figure 6.15 Item Category window

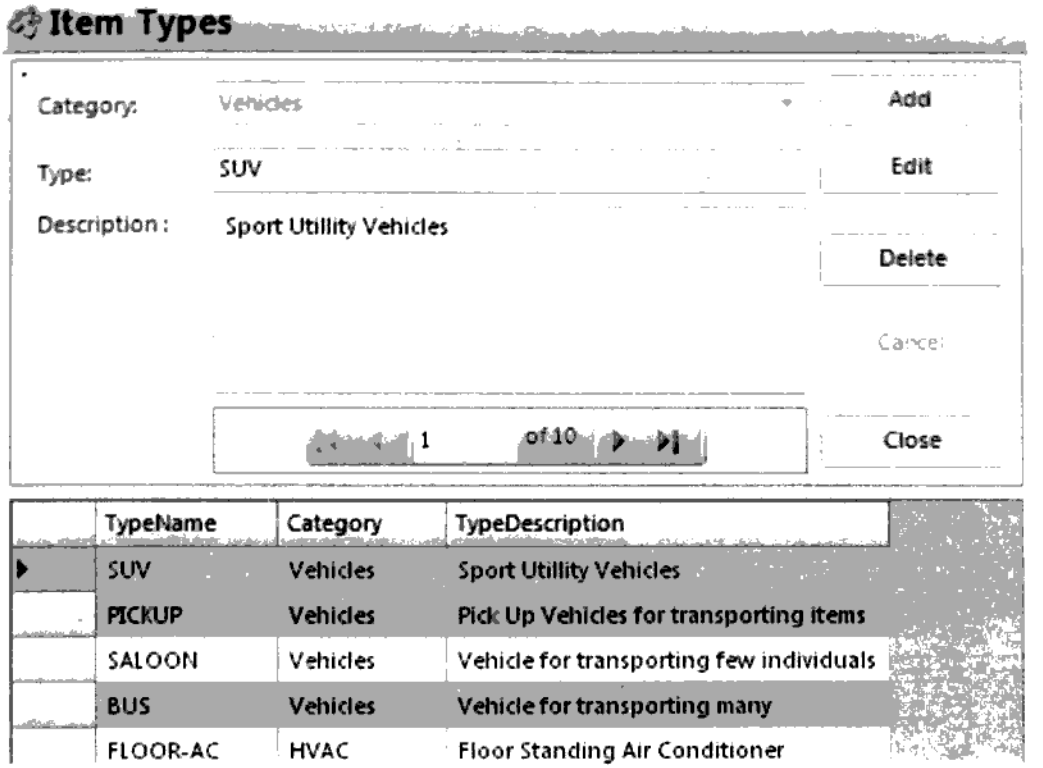


Figure 6.16 Item Type window

Item Class/Group: The Item Class combines the 'Category', 'Type' and 'Brand' into a group item. It also classifies the group item into Models. This group item is what individual assets derive from. For example, a group item Rav4 is of category vehicle, of type SUV and brand Toyota *(See Figure 6.17)*

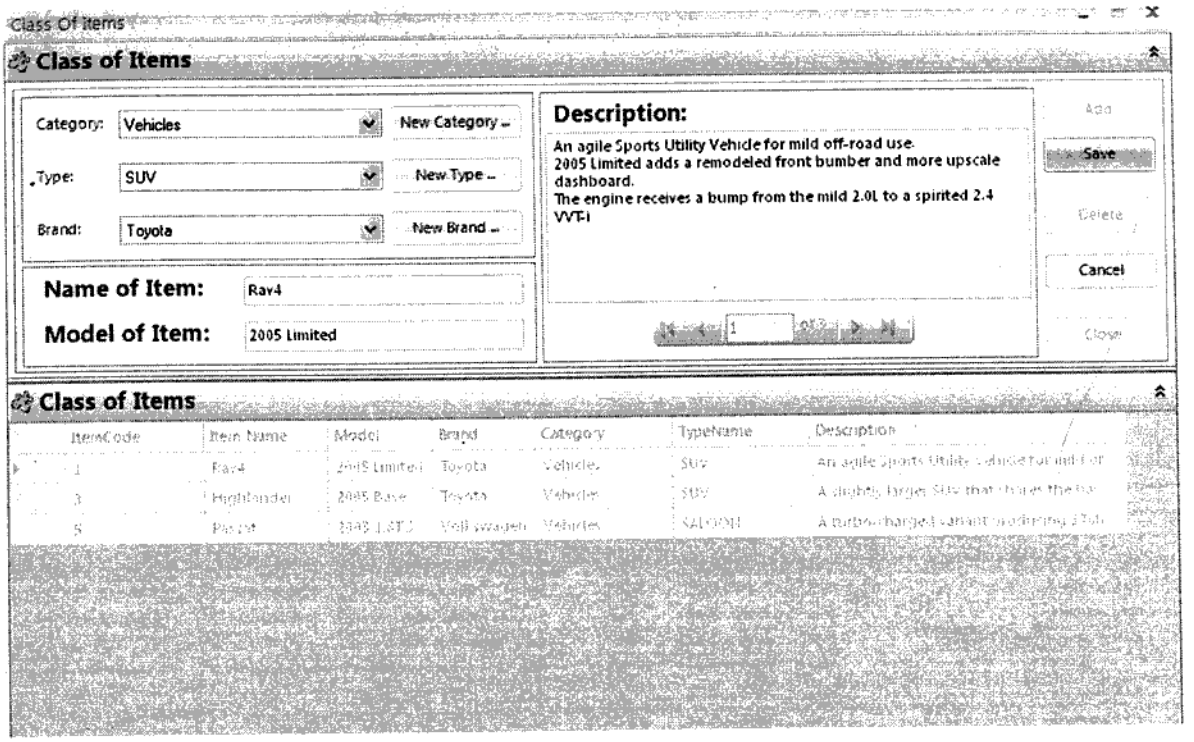


Figure 6.17 Item Class window

Item Specification block

The Item Specification module (Figure 6.18) is the only component of this block. It is used for cataloguing of assets. Details of the asset are entered into the system by choosing the correct item class and assigning its asset identification number and/or serial number. The Item Specification module is an inventory of asset. Some details include the warranty expiration date, location, supplier and remarks about the age of the item.

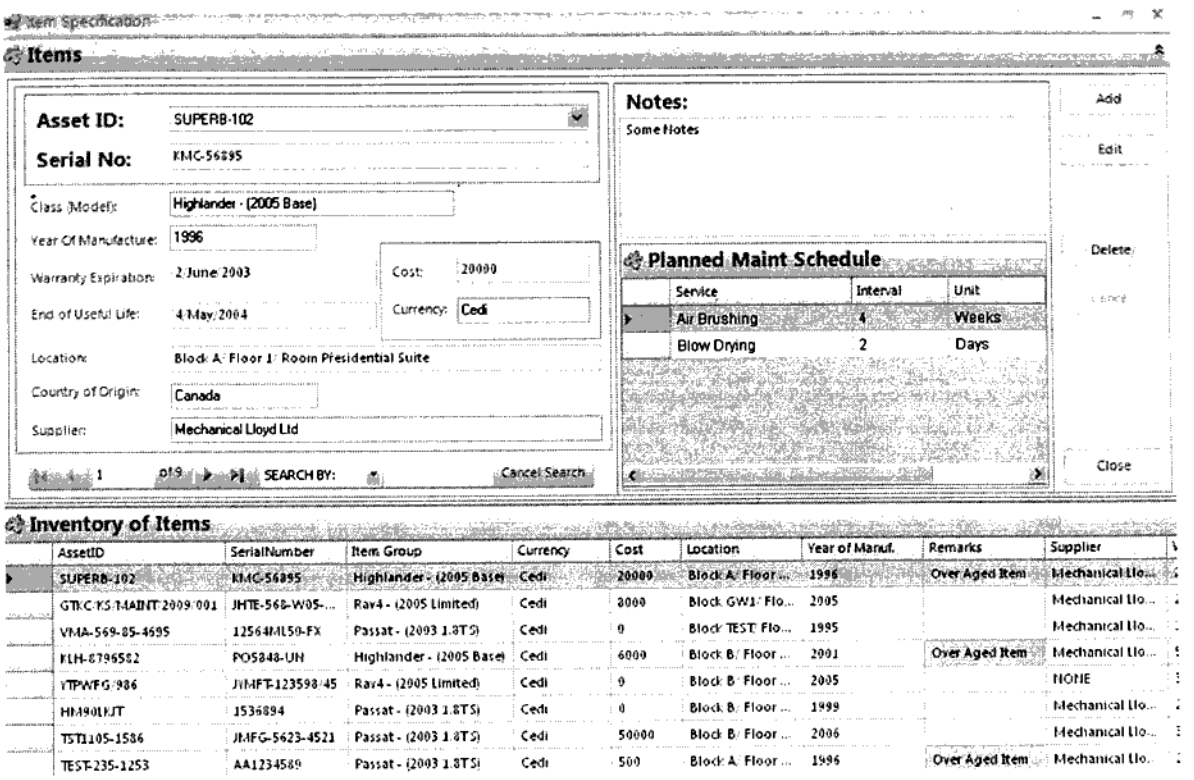


Figure 6.18 Item Specification window

Maintenance block

The Maintenance block brings together all activities related to maintenance. It consists of an Individual Scheduler module, a Group Scheduler module, a Planned Maintenance Manager module, Unplanned Maintenance module and Maintenance reports modules. The Maintenance block constitutes the Maintenance Manager.

Individual and Group Schedulers: These are responsible for assigning planned maintenance schedules of various services to assets either individually or as groups *(See Figures 6.19, 6.20).*

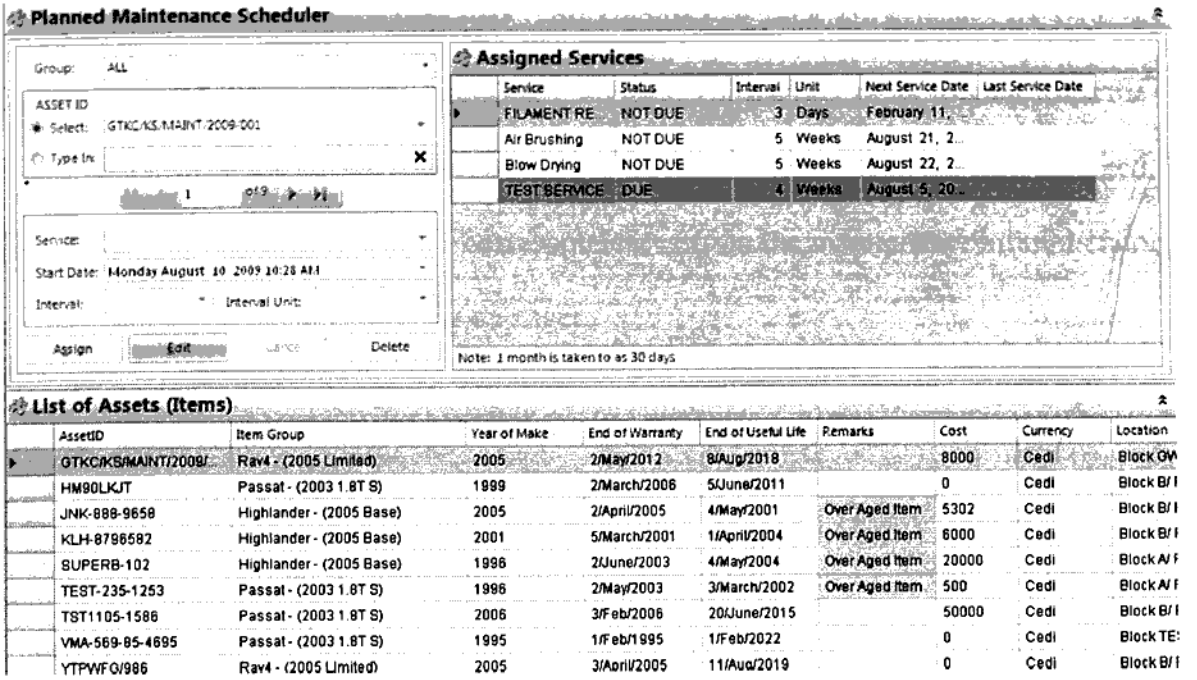


Figure 6.19 Individual Scheduler

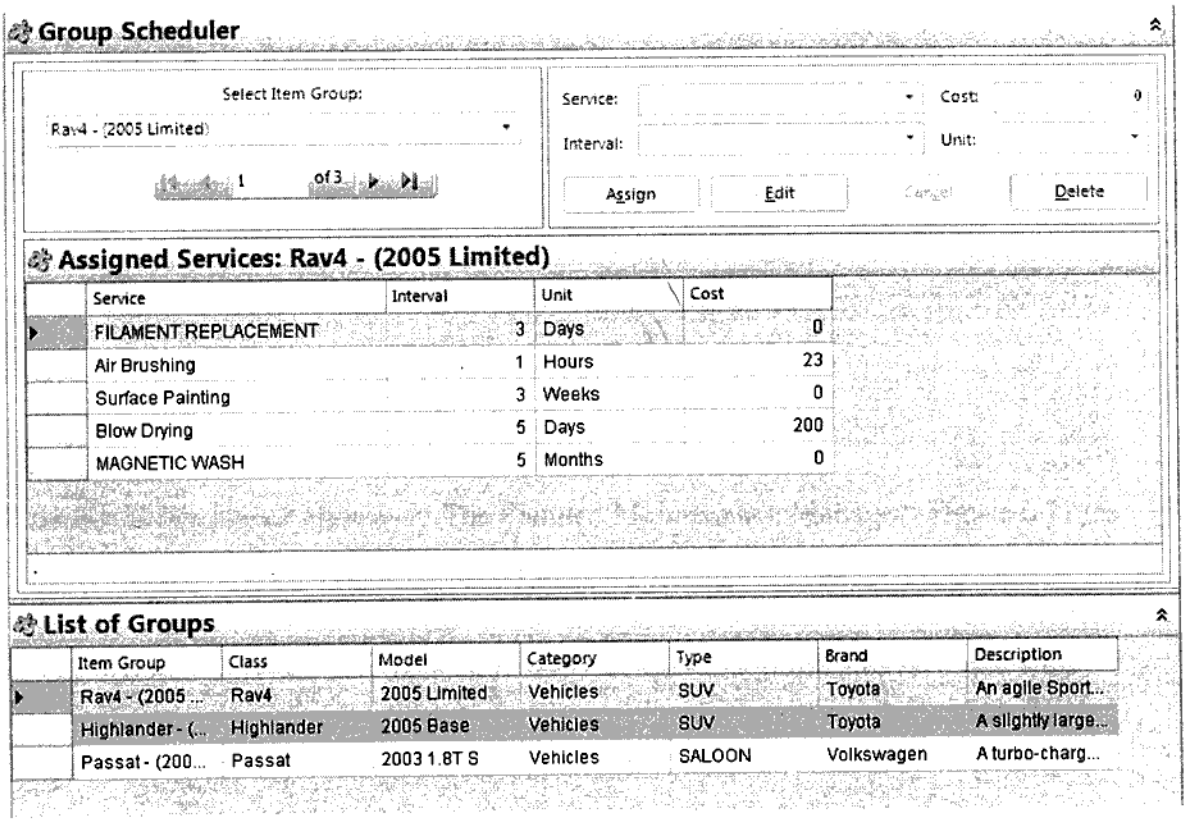


Figure 6.20 Group Scheduler

Unplanned Maintenance Manager: This module (Figure 6.21) manages the requests that fall outside planned maintenance activities. The user requests coming from the Unplanned Maintenance Client is received by this manager. Staff are then assigned to the task and dispatched. All correspondences from the maintenance administrator to the user is also done via this manager.

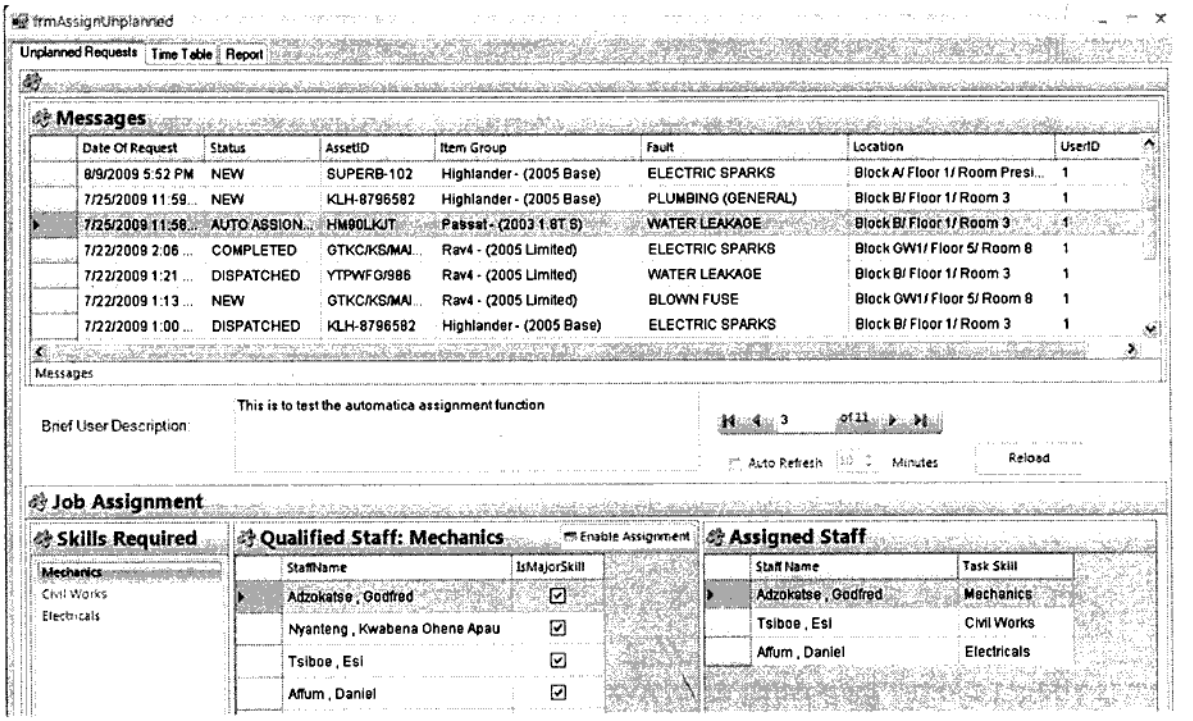


Figure 6.21 Unplanned Maintenance Manager

Planned Maintenance Manager: The Planned Maintenance Manager (Figure 6.22) is similar to the Unplanned Maintenance Manager with the exception that it takes its input from a schedule instead of a request. It also manages assignment of staff and staff dispatches.

It has a job assignment tab for assigning tasks to staff, a planned schedule tab for listing the planned schedules of assets, a list of all dispatched jobs, a service work-plan tab and a time table tab for displaying the time tables of staff.

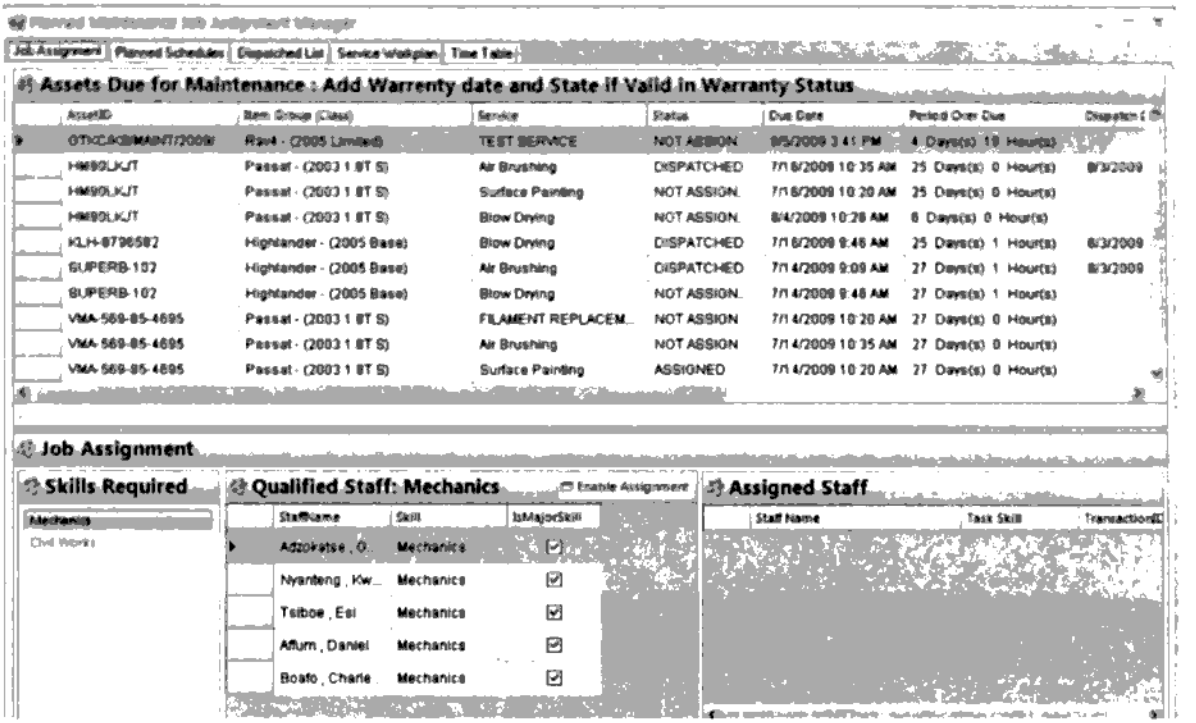


Figure 6.22 Planned Maintenance Manager

Reports: The Maintenance block contains a module for generating various reports about maintenance activities. These reports includes a maintenance summary that summarises all maintenance activities and the costs associated with such activities, a maintenance history of assets, history of staff assignment and a dispatch list etc. Some of these reports can be queried to using the criteria of interest. The reports can be exported to PDF or Excel Formats

Figure 6.23 Sample Report window

6.5 Sample Case Studies

6.5.1 Unplanned Maintenance Requests

A cleaner in one of the guest rooms notices that there are some electrical sparks coming from a television set in the room. The cleaner puts off the appliance and decided to report the incident using the Unplanned Maintenance Client.

Logging into the system brings the Unplanned Maintenance Client Message window. Notice that there are no messages in the Message window (Figure 6.24).

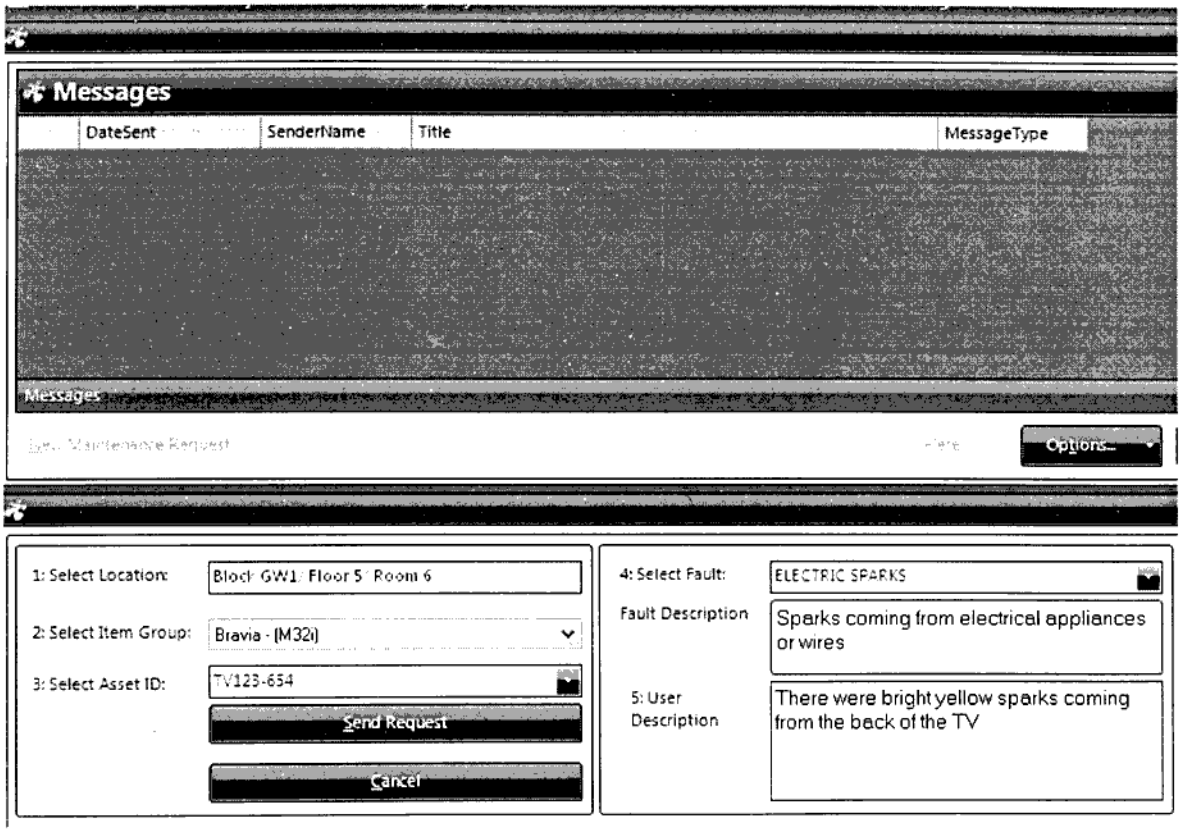


Figure 6.24 The Unplanned Maintenance Message window

On submission of the fault, a message is automatically sent to the cleaner of the receipt of message as shown in Figure 6.25.

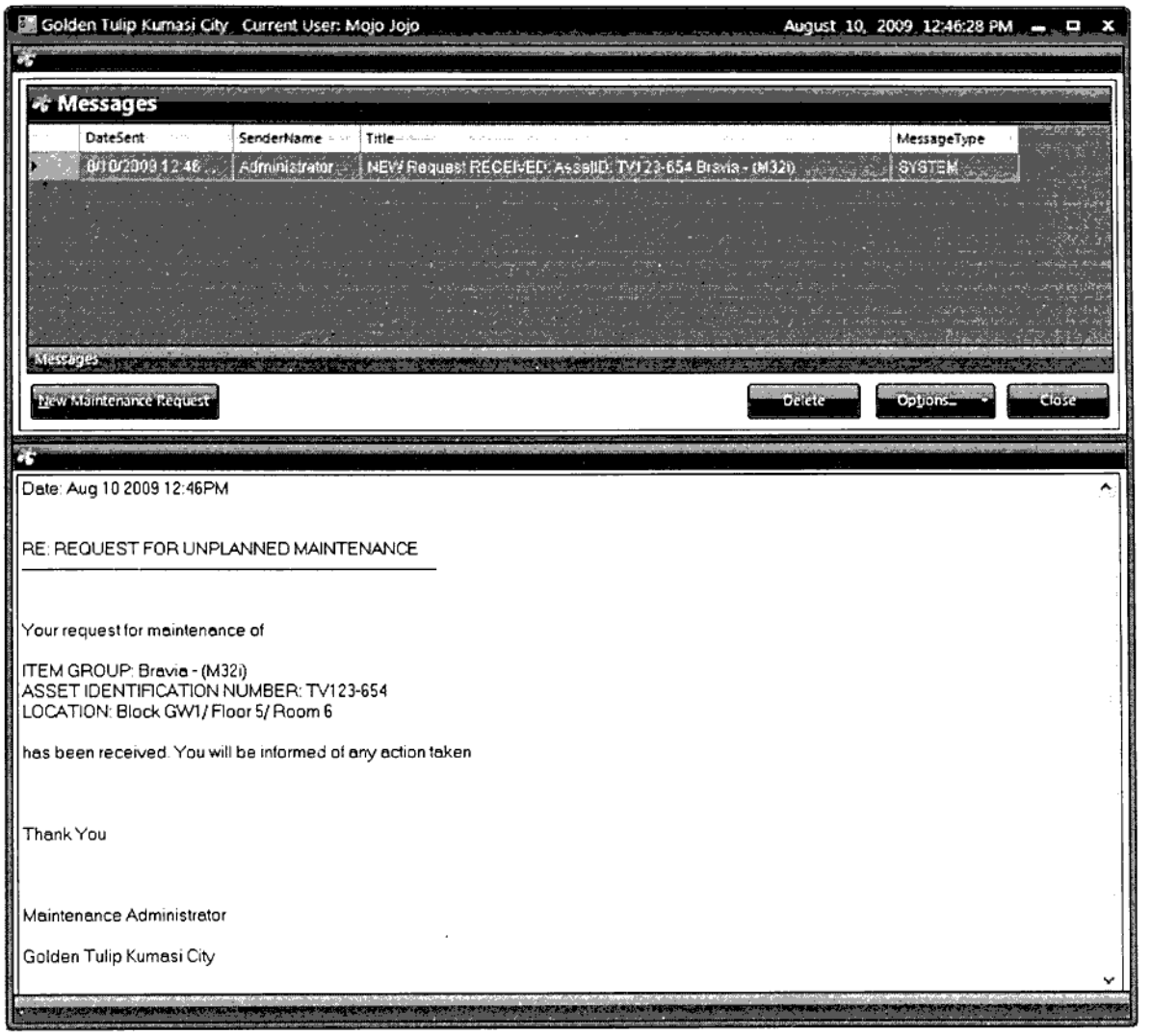


Figure 6.25 Message window displaying acknowledgement

The maintenance administrator, upon receipt of the request (Figure 6.26), reviews a list of suggestions by the system on competent staff for the job. He assigns the staff and dispatches them to the task (Figure 6.27, 6.29).

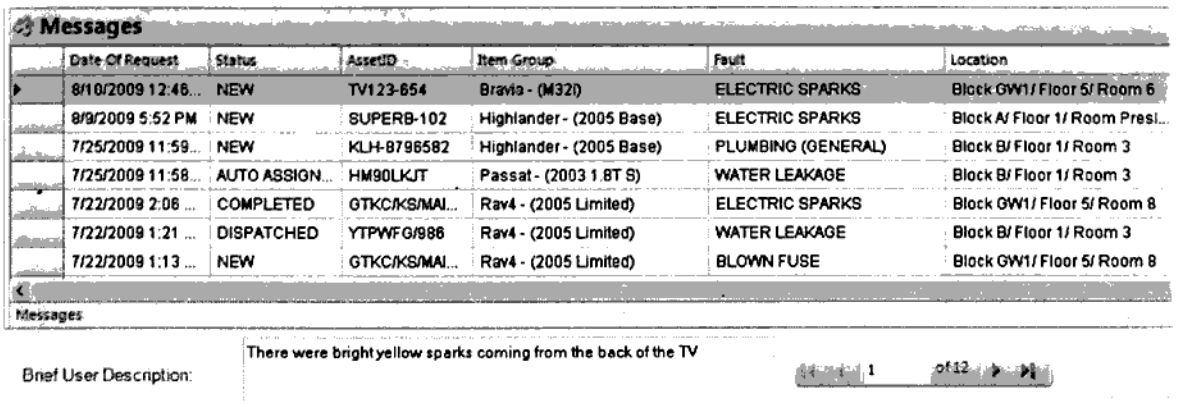


Figure 6.26 Unplanned Maintenance Manager's view of the new request

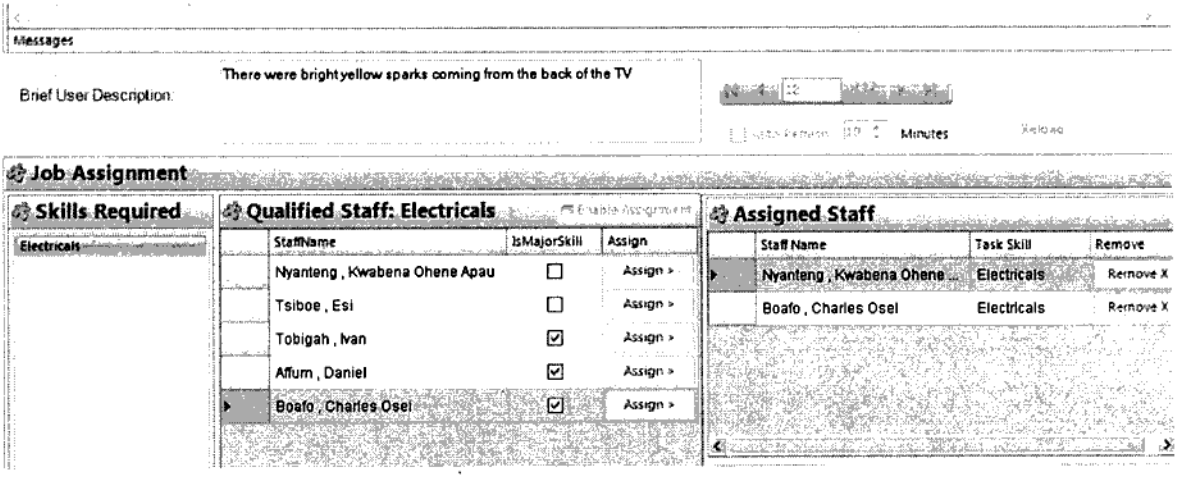


Figure 6.27 A view of competent (left) and staff assigned (right)

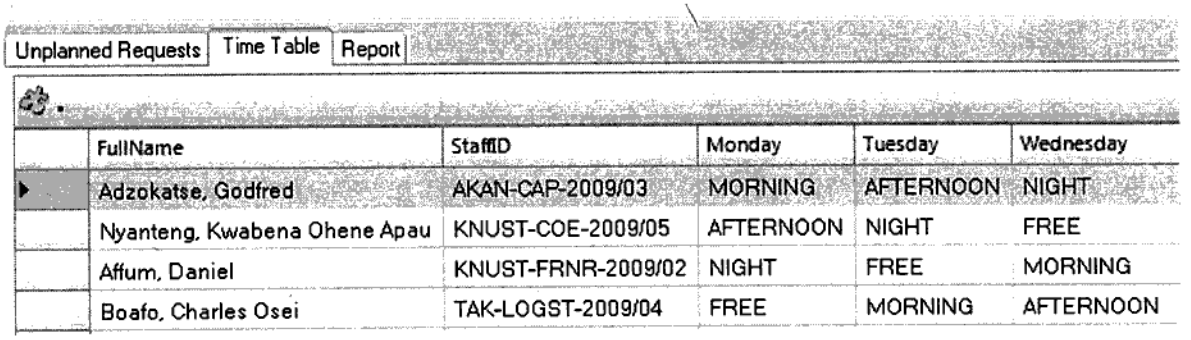
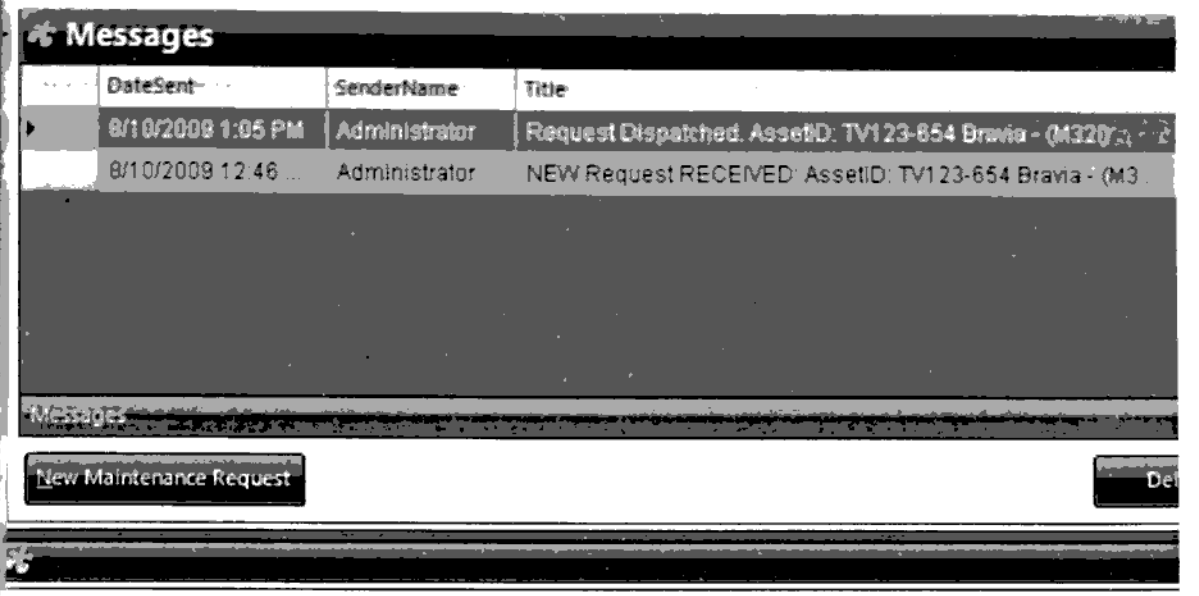


Figure 6.28 A view of the work-shift of staff



Date: Aug 10 2009 1:05PM

Staff haave been dispatched for the maintenance of

ITEM GROUP: Bravia - (M32i)
ASSET IDENTIFICATION NUMBER: TV123-654
LOCATION: Block GW1/ Floor 5/ Room 6

. You will be informed upon completetion

Thank You

Maintenance Administrator

Figure 6.29 Cleaner's notification of dispatch

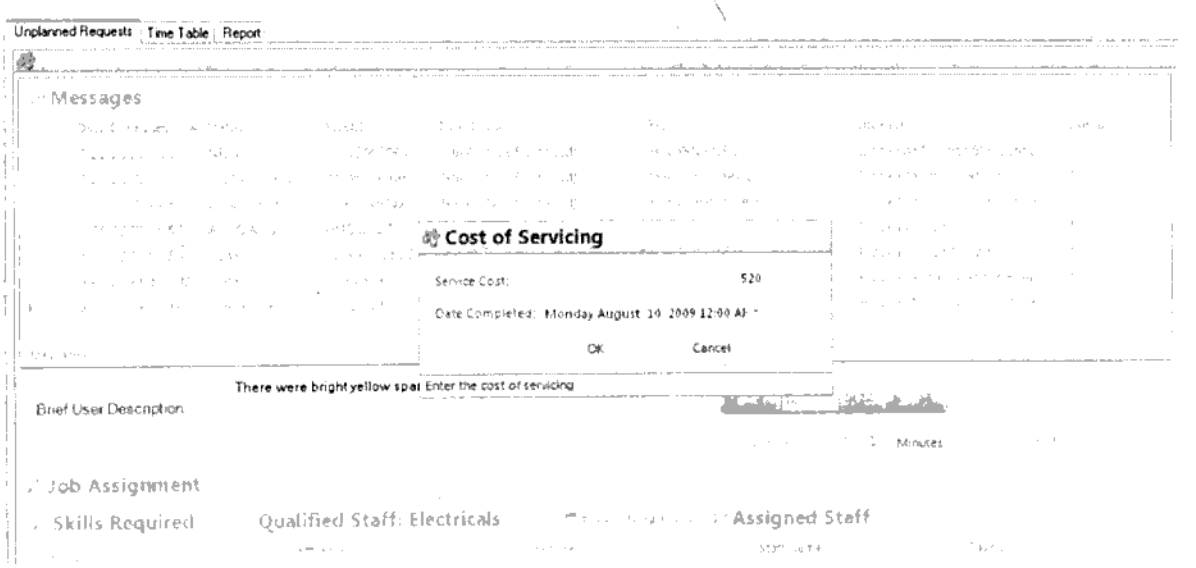


Figure 6.30 Finalizing the job when completed

Upon completion of the task, the maintenance administrator marks the job as completed with the cost involved and the date of completion as shown in Figure 6.30. The cleaner is then notified of the completion of the task (Figure 6.31).

Figure 6.31 Notification of task completion

6.5.2 Planned Maintenance Schedule

The maintenance administrator wants to schedule an item for regular preventative maintenance that will recur hourly beginning from the time of schedule. He logs into the system and starts up the Individual Planned Maintenance Scheduler (Figure 6.32).

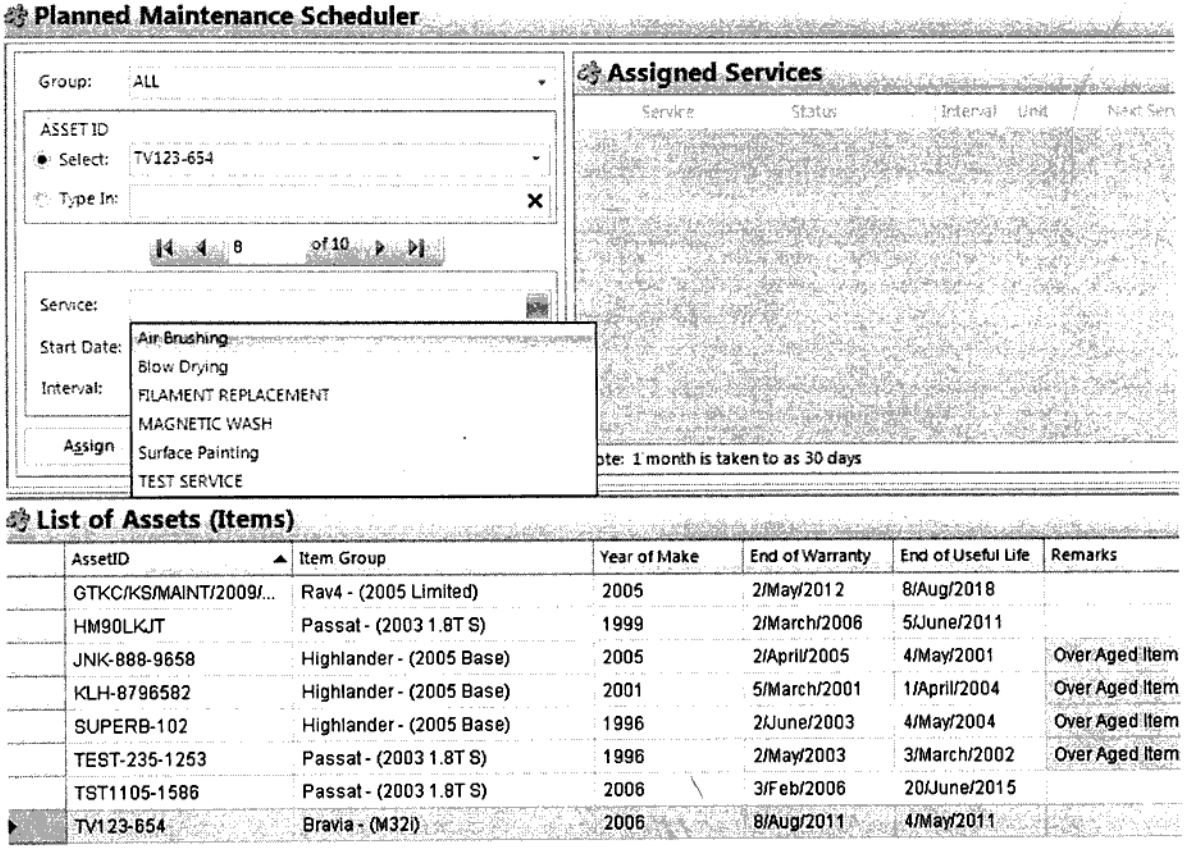


Figure 6.32 Selection of Service for scheduling

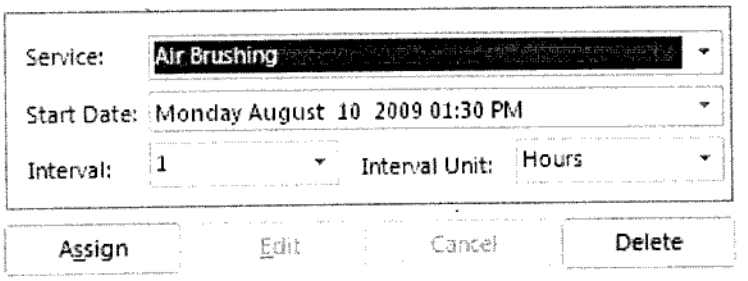


Figure 6.33 Selection of time interval for servicing

A service interval is selected by the administrator and assigned (Figure 6.33)

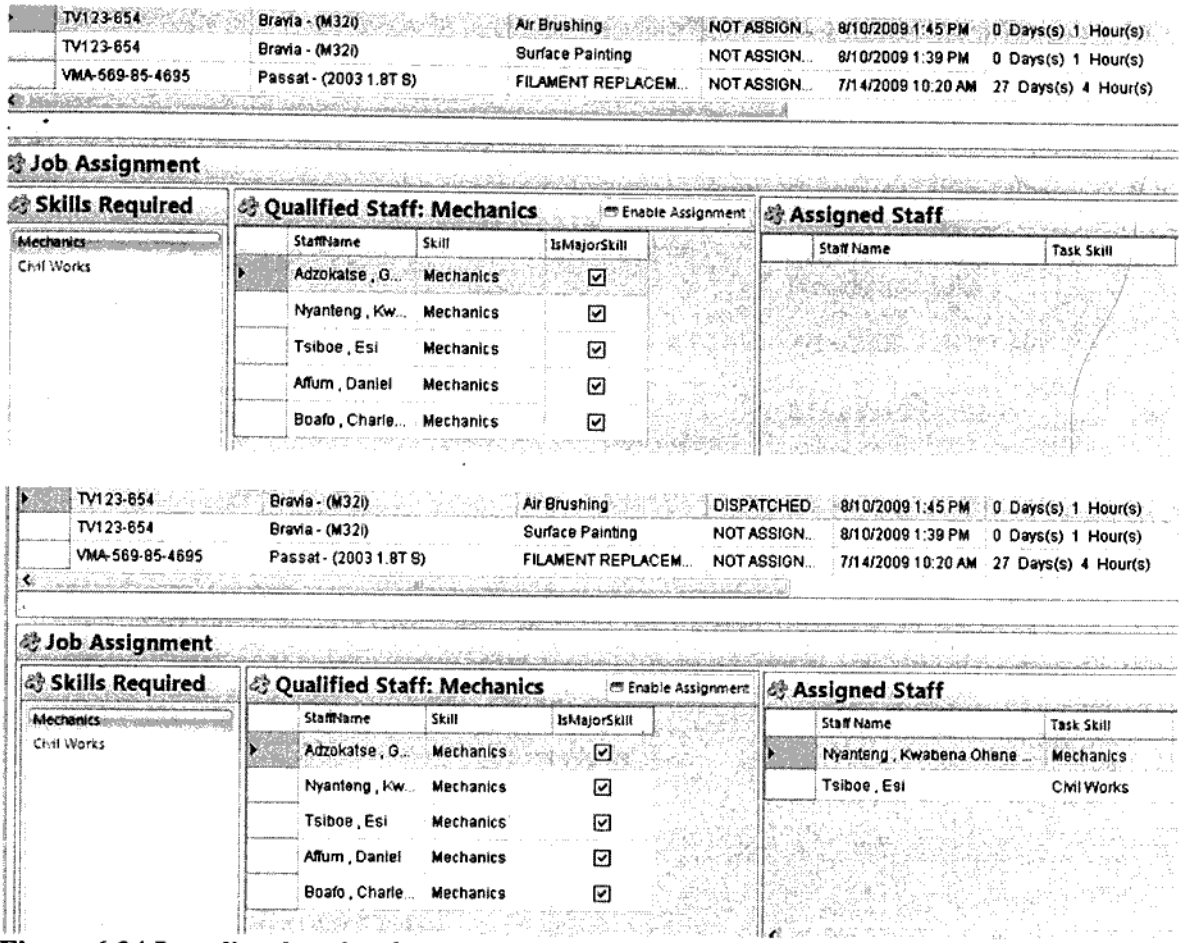


Figure 6.34 Item listed as due for service(top) , dispatched staff (bottom)

The staff assigned to the task are dispatched and issued out a dispatch form (Figure 6.35) and a work plan (Figure 6.36).

On completion, it is marked as completed (Figure 6.37) and placed in the maintenance history table (Figure 6.56.38).

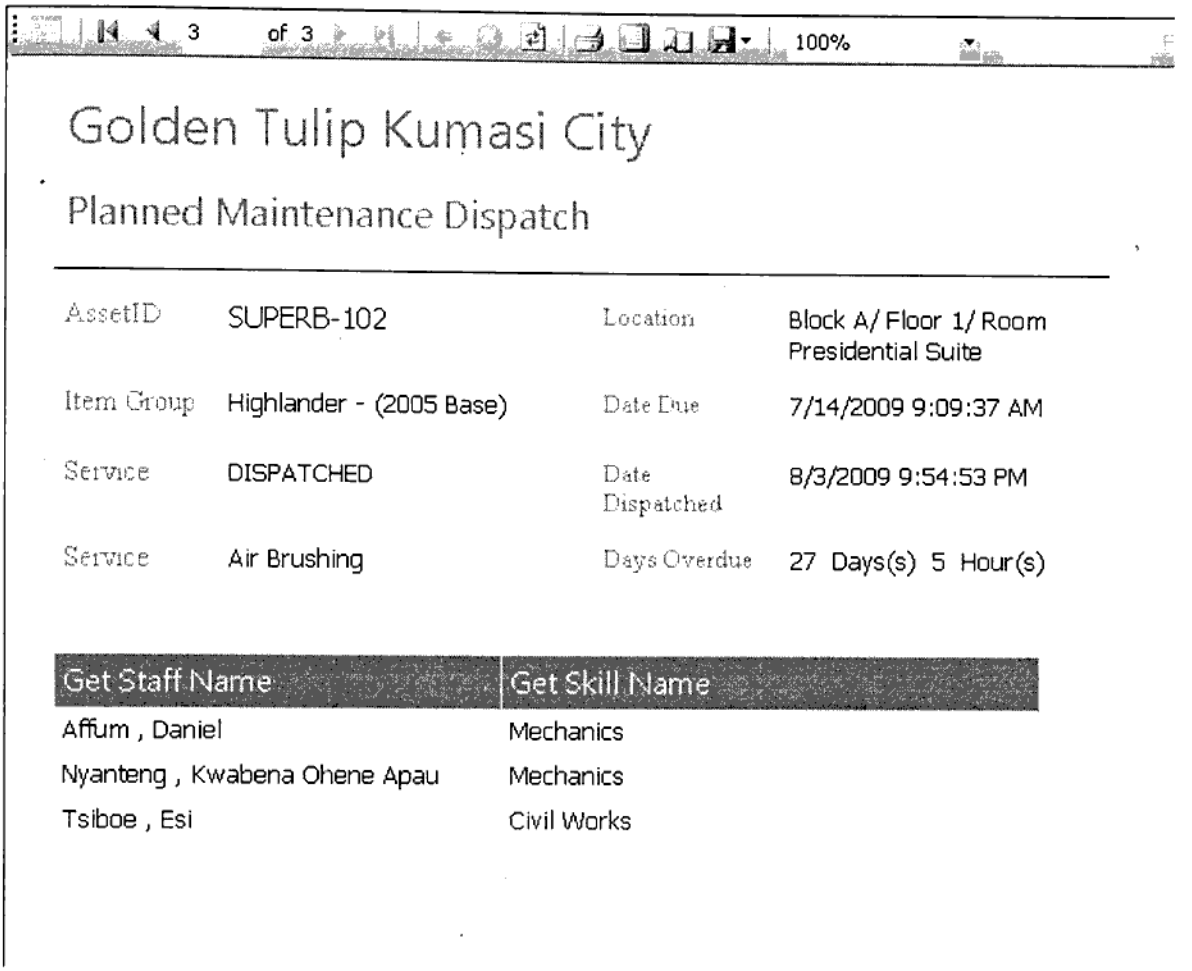


Figure 6.35 Sample Dispatch form

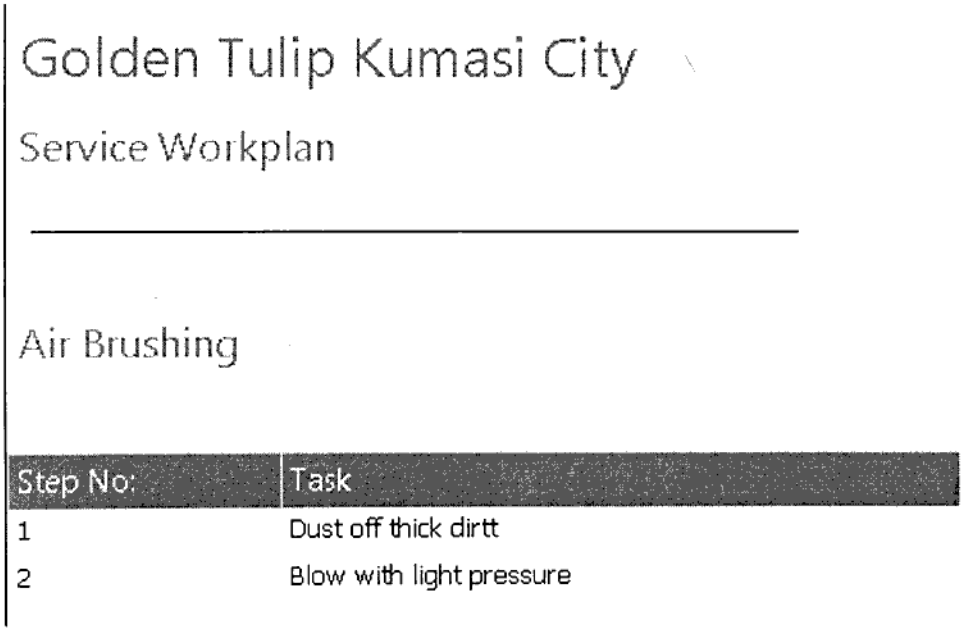


Figure 6.36 Service Work-plan

Golden Tulip Kumasi City

Asset Maintenance History

History ID	20	Date of Entry	8/10/2009 2:51:50 PM
Asset ID	TV123-654	Type	PLANNED
Type	Bravia - (M32i)	Cost	50
Fault/Service	Air Brushing	Date of Completion	8/10/2009 2:48:46 PM
Fault/Service	8/10/2009 2:44:00 PM		

SKill Name	Staff Assigned
Mechanics	Nyanteng , Kwabena Ohene Apau
Civil Works	Tsiboe , Esi

Figure 6.37 Report on completed planned task

Golden Tulip Kumasi City

Maintenance Summary

Asset ID	Type	Planned Count	UnPlanned Count	Total Service Count	Plannned Cost	Unplanned Cost	Total Cost
GTKC/KS/MAINT/2009/001	Rav4 - (2005 Limited)	4	1	5	1344	300	1644
HM90LKJT	Passat - (2003 1.8T S)	7		7	1974		1974
JNK-888-9658	Highlander - (2005 Base)			0			0
KLH-8796582	Highlander - (2005 Base)	2		2	445		445
SUPERB-102	Highlander - (2005 Base)			0			0
TEST-235-1253	Passat - (2003 1.8T S)			0			0
TST1105-1586	Passat - (2003 1.8T S)			0			0
TV123-654	Bravia - (M32i)	1	1	2	50	520	570
VMA-569-85-4695	Passat - (2003 1.8T S)			0			0
YTPWFG/986	Rav4 - (2005 Limited)			0			0

Figure 6.38 Maintenance Summary Statistics

7 CONCLUSIONS AND RECOMMENDATIONS

7.1 Conclusion

Maintaining a vast number of assets on a regular basis forms a huge challenge to many organisations especially, those in the hospitality sector. The GT Maintenance Management Software has addressed some of these challenges. It addresses asset locations, asset classification and categorisation.

It also addresses staff information, qualifications and work-shifts.

It enables users place unplanned requests and provides an interface for tracking the status of those requests.

Planned maintenance scheduling and staff suggestions are also tackled.

By taking care of some of these issues, the software will assist organisations within the hospitality industry to give better quality of service to their customers. It will also enable the avoidance of many expensive repairs due to adherence to the timely preventative maintenance practices.

7.2 Recommendations

Software management systems are designed to facilitate work. For the GTMMS to be successful, it requires the devotion of its user. The software can only prompt of an impending maintenance activity but it is up to the human factor to ensure that such suggestions are adhered to.

Manual Checks on the maintenance schedules of assets should, occasionally, be done to verify that the results of the system tallies with what pertains on the ground.

Problematic equipment should be considered for replacement to minimise servicing costs.

In the future, the Unplanned Maintenance Client should be ported to a web based interface to make it more accessible. This is because the current version of the Unplanned Maintenance Client is required to be installed on the requesting computer. A web based approach will eliminate the need for such an installation thereby making the Client accessible virtually on any computer connected to the network.

8 References

Ahmed, M., Garrett, C., & Faircloth, J. (2002). *ASP.NET Web Developer's Guide.* Sygress Publishing Inc.

Bryla, B. (2003). *Oracle 9i DBA Jump Start.* Sybex Inc.

Buzzle.com. (2008, 8 5). Retrieved July 20, 2009, from Advantages of Relational Databases: http://www.buzzle.com/articles/advantages-of-relational-databases.html

Clever Solutions. (2009). *Clever CMMS user friendly preventative maintenance software.* Retrieved 7 12, 2009, from Clever CMMS: The Maintenace Management Solution: http://www.clevercmms.com

CWORKS SYSTEMS. (2009). *CWorks: lowcost and free.* Retrieved 7 13, 2009, from CWorks CMMS: http://cworks.com.my

Dewson, R. (2006). *Beginning SQL Server 2005 for Developers.* Apress.

Encarta. (2009). Retrieved July 18, 2009, from http://encarta.msn.com: http://encarta.msn.com/dictionary_1861628101/management.html

IDCON. (2009). Retrieved July 15, 2009, from Reliability and Maintenance Definitions: http://www.idcon.com/maintenance-definition-M.htm

Kauffman, J., Claudio, F., Chinnathampi (Das), J., Greenwood, J., & Matsik, B. (2002). *ASP.Net Databases Using VB.NET.* Wrox Press Ltd.

McManus, J. P., & Goldstein, J. (2003). *Database Access with Visual Basic .NET.* Addison Wesley.

Petroutsos, E., & Bilgin, A. (2002). *Mastering Visula Basic .Net Database Programming.* Sybex Inc.

Price, J. (2003). *Mastering C# Database Programming.* Sybex INc.

(2005). Understanding n-tier Design. In B. Sempf, *Visual Basic 2005 for Dummies* (p. 73). Wiley Publishing Inc.

Shead, M. (2007, January 30). *http://www.leadership501.com.* Retrieved July 19, 2009, from Leadership501: http://www.leadership501.com/definition-of-management/21/

Smartware Group. (2008). *Bigfoot CMMS.* Retrieved 7 13, 2009, from Bigfoot Maintenace Software: http://bigfootcmms.com

Wikipedia. (2009, July 21). Retrieved July 23, 2009, from Management System: http://en.wikipedia.org/wiki/Management_system

Wikipedia. (2009, July 22). Retrieved July 22, 2009, from Computerized Maintenance Management System: http://en.wikipedia.org/wiki/CMMS

Wikipedia. (2009, July 22). Retrieved July 23, 2009, from Software Development Process: http://en.wikipedia.org/wiki/Software_development_process

Wikipedia. (2009, July 20). Retrieved July 23, 2009, from CodeGear Delphi: http://en.wikipedia.org/wiki/Borland_Delphi

Wikipedia. (2009, July 19). Retrieved July 19, 2009, from Visual Basic.Net: http://en.wikipedia.org/wiki/Visual_Basic.Net

Wikipedia.org. (2009, July 25). Retrieved July 22, 2009, from Programming Languages: http://en.wikipedia.org/wiki/Programming_languages

Wiktionary. (2009, July 18). Retrieved July 20, 2009, from Wiktionary.org: http://en.wiktionary.org/wiki/maintenance

Wkipedia. (2009, July 26). Retrieved July 27, 2009, from Database Normalization: http://en.wikipedia.org/wiki/Normalization_(database)

Appendices

Appendix A: Database Structure.

The following script is a description of the database structure

```
USE [master]
GO
/****** Object:  Database [GoldenData]    Script Date: 08/10/2009
15:05:42 ******/
IF NOT EXISTS (SELECT name FROM sys.databases WHERE name =
N'GoldenData')
BEGIN
CREATE DATABASE [GoldenData] ON  PRIMARY
( NAME = N'GoldenData', FILENAME = N'D:\My Thesis\GoldenData.mdf' ,
SIZE = 4096KB , MAXSIZE = UNLIMITED, FILEGROWTH = 1024KB )
 LOG ON
( NAME = N'GoldenData_log', FILENAME = N'D:\My
Thesis\GoldenData_log.LDF' , SIZE = 1024KB , MAXSIZE = UNLIMITED,
FILEGROWTH = 10%)
END

GO
EXEC dbo.sp_dbcmptlevel @dbname=N'GoldenData', @new_cmptlevel=90
GO
IF (1 = FULLTEXTSERVICEPROPERTY('IsFullTextInstalled'))
begin
EXEC [GoldenData].[dbo].[sp_fulltext_database] @action = 'enable'
end
GO
GO
ALTER DATABASE [GoldenData] SET DB_CHAINING OFF
USE [GoldenData]
GO
/****** Object:  Table [dbo].[Category]    Script Date: 08/10/2009
15:05:45 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[Category]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[Category](
      [CategoryID] [int] IDENTITY(1,1) NOT NULL,
      [CategoryName] [nvarchar](50) NOT NULL,
      [CategoryDescription] [nvarchar](500) NULL,
 CONSTRAINT [PK_Category] PRIMARY KEY CLUSTERED
(
      [CategoryID] ASC
```

```
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
END
GO

/****** Object:  Index [IX_Category]    Script Date: 08/10/2009
15:05:45 ******/
IF NOT EXISTS (SELECT * FROM sys.indexes WHERE object_id =
OBJECT_ID(N'[dbo].[Category]') AND name = N'IX_Category')
CREATE UNIQUE NONCLUSTERED INDEX [IX_Category] ON [dbo].[Category]
(
      [CategoryName] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[FormsInUse]    Script Date: 08/10/2009
15:05:45 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[FormsInUse]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[FormsInUse](
      [FormName] [nvarchar](50) NOT NULL,
      [InUse] [bit] NOT NULL,
      [UserName] [nvarchar](50) NULL,
      [TimeStarted] [datetime] NULL,
 CONSTRAINT [PK_FormsInUse] PRIMARY KEY CLUSTERED
(
      [FormName] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
END
GO
/****** Object:  Table [dbo].[Brand]    Script Date: 08/10/2009
15:05:45 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[Brand]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[Brand](
      [BrandCode] [int] IDENTITY(1,1) NOT NULL,
      [BrandName] [nvarchar](50) NOT NULL,
      [Description] [nvarchar](500) NULL,
 CONSTRAINT [PK_Brand] PRIMARY KEY CLUSTERED
(
      [BrandCode] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
END
GO
```

```
/****** Object:  Index [IX_Brand]    Script Date: 08/10/2009 15:05:45
******/
IF NOT EXISTS (SELECT * FROM sys.indexes WHERE object_id =
OBJECT_ID(N'[dbo].[Brand]') AND name = N'IX_Brand')
CREATE UNIQUE NONCLUSTERED INDEX [IX_Brand] ON [dbo].[Brand]
(
      [BrandName] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Shift]    Script Date: 08/10/2009
15:05:45 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[Shift]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[Shift](
      [ShiftCode] [nvarchar](50) NOT NULL,
      [ShiftDescription] [nvarchar](500) NULL,
      [BeginTime] [datetime] NOT NULL,
      [EndTime] [datetime] NOT NULL,
      [AutoID] [int] IDENTITY(1,1) NOT NULL,
      [ComputedBeginTime]  AS (right([BeginTime],(7))),
      [ComputedEndTime]  AS (right([EndTime],(7))),
 CONSTRAINT [PK_Shift] PRIMARY KEY CLUSTERED
(
      [ShiftCode] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
END
GO

/****** Object:  Index [IX_Shift]    Script Date: 08/10/2009 15:05:45
******/
IF NOT EXISTS (SELECT * FROM sys.indexes WHERE object_id =
OBJECT_ID(N'[dbo].[Shift]') AND name = N'IX_Shift')
CREATE NONCLUSTERED INDEX [IX_Shift] ON [dbo].[Shift]
(
      [BeginTime] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
GO

/****** Object:  Trigger [dbo].[DeleteShift]    Script Date: 08/10/2009
15:05:45 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

IF NOT EXISTS (SELECT * FROM sys.triggers WHERE object_id =
OBJECT_ID(N'[dbo].[DeleteShift]'))
EXEC dbo.sp_executesql @statement = N'CREATE TRIGGER
[dbo].[DeleteShift] on [dbo].[Shift]
for delete
```

```
as
declare @CodeDeleted nvarchar(50)

select @CodeDeleted = ShiftCode from deleted

update StaffShift set StaffShift.MondayShiftCode = ''FREE'' where
StaffShift.MondayShiftCode = @CodeDeleted
update StaffShift set StaffShift.TuesdayShiftCode = ''FREE'' where
StaffShift.TuesdayShiftCode = @CodeDeleted
update StaffShift set StaffShift.WednesdayShiftCode = ''FREE'' where
StaffShift.WednesdayShiftCode = @CodeDeleted
update StaffShift set StaffShift.ThursdayShiftCode = ''FREE'' where
StaffShift.ThursdayShiftCode = @CodeDeleted
update StaffShift set StaffShift.FridayShiftCode = ''FREE'' where
StaffShift.FridayShiftCode = @CodeDeleted
update StaffShift set StaffShift.SaturdayShiftCode = ''FREE'' where
StaffShift.SaturdayShiftCode = @CodeDeleted
update StaffShift set StaffShift.SundayShiftCode = ''FREE'' where
StaffShift.SundayShiftCode = @CodeDeleted
'
GO

/****** Object:  Trigger [dbo].[Updatehift]    Script Date: 08/10/2009
15:05:45 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

IF NOT EXISTS (SELECT * FROM sys.triggers WHERE object_id =
OBJECT_ID(N'[dbo].[Updatehift]'))
EXEC dbo.sp_executesql @statement = N'CREATE TRIGGER [dbo].[Updatehift]
on [dbo].[Shift]
for update

as
declare @CodeDeleted nvarchar(50), @NewCode nvarchar(50)

select @CodeDeleted = ShiftCode from deleted
select @NewCode = ShiftCode from inserted


update StaffShift set StaffShift.SundayShiftCode = @NewCode where
StaffShift.SundayShiftCode = @CodeDeleted
'
GO
IF NOT EXISTS (SELECT * FROM
::fn_listextendedproperty(N'MS_Description' , N'SCHEMA',N'dbo',
N'TABLE',N'Shift', N'COLUMN',N'ComputedBeginTime'))
EXEC sys.sp_addextendedproperty @name=N'MS_Description',
@value=N'Contains the text representtion of the extracted commencement
time' , @level0type=N'SCHEMA',@level0name=N'dbo',
@level1type=N'TABLE',@level1name=N'Shift',
@level2type=N'COLUMN',@level2name=N'ComputedBeginTime'
GO
```

```
/****** Object:  Table [dbo].[StoresTransactionTable]    Script Date:
08/10/2009 15:05:46 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[StoresTransactionTable]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[StoresTransactionTable](
      [TransactionID] [int] IDENTITY(1,1) NOT NULL,
      [TransactionDate] [datetime] NOT NULL,
      [ItemCode] [nvarchar](50) NOT NULL,
      [ItemName] [nvarchar](50) NOT NULL,
      [ItemCategoryName] [nvarchar](50) NULL,
      [ItemCategoryCode] [nvarchar](50) NULL,
      [InitialQuantiyAvailable] [int] NULL,
      [QuantityTaken] [int] NULL,
      [QuantityRemaining] [int] NULL,
      [NameOfSupplier] [nvarchar](50) NULL,
      [NameOfRequester] [nvarchar](50) NULL,
 CONSTRAINT [PK_StoresTransactionTable] PRIMARY KEY CLUSTERED
(
      [TransactionID] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
END
GO
/****** Object:  Table [dbo].[Skill]    Script Date: 08/10/2009
15:05:46 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[Skill]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[Skill](
      [SkillID] [int] IDENTITY(1,1) NOT NULL,
      [SkillName] [nvarchar](50) NOT NULL,
      [SkillDescription] [nvarchar](500) NULL,
 CONSTRAINT [PK_Skill] PRIMARY KEY CLUSTERED
(
      [SkillID] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
END
GO
/****** Object:  UserDefinedFunction [dbo].[TimeOnly]    Script Date:
08/10/2009 15:05:46 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[TimeOnly]') AND type in (N'FN', N'IF', N'TF', N'FS',
N'FT'))
```

```
BEGIN
/****** Object:  Table [dbo].[WorkStatus]    Script Date: 08/10/2009
15:05:46 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[WorkStatus]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[WorkStatus](
	[AutoID] [int] IDENTITY(1,1) NOT NULL,
	[SenderID] [nvarchar](50) NULL,
	[ReceipientID] [nvarchar](50) NULL,
	[Title] [nvarchar](100) NULL,
	[MessageBody] [nvarchar](800) NULL,
	[DateSent] [datetime] NULL,
	[MessageType] [nvarchar](50) NULL,
	[SenderName] [nvarchar](50) NULL,
	[ReceipientName] [nvarchar](50) NULL,
 CONSTRAINT [PK_WorkStatus] PRIMARY KEY CLUSTERED
(
	[AutoID] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
END
GO
/****** Object:  Table [dbo].[CategoryTable]    Script Date: 08/10/2009
15:05:46 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id
OBJECT_ID(N'[dbo].[CategoryTable]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[CategoryTable](
	[CategoryCode] [int] IDENTITY(1,1) NOT NULL,
	[CategoryName] [nvarchar](50) NOT NULL,
 CONSTRAINT [PK_CategoryTable] PRIMARY KEY CLUSTERED
(
	[CategoryCode] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
END
GO
IF NOT EXISTS (SELECT * FROM
::fn_listextendedproperty(N'MS_Description' , N'SCHEMA',N'dbo',
N'TABLE',N'CategoryTable', N'COLUMN',N'CategoryCode'))
EXEC sys.sp_addextendedproperty @name=N'MS_Description', @value=N'An
auto generated index' , @level0type=N'SCHEMA',@level0name=N'dbo',
@level1type=N'TABLE',@level1name=N'CategoryTable',
@level2type=N'COLUMN',@level2name=N'CategoryCode'
GO
/****** Object:  Table [dbo].[MaintenanceHistory]    Script Date:
08/10/2009 15:05:46 ******/
SET ANSI_NULLS ON
```

```
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[MaintenanceHistory]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[MaintenanceHistory](
	[HistoryID] [int] IDENTITY(1,1) NOT NULL,
	[DateOfEntry] [datetime] NULL,
	[ItemClassName] [nvarchar](50) NULL,
	[AssetID] [nvarchar](50) NOT NULL,
	[ServiceID] [nvarchar](50) NULL,
	[ServiceOrFaultName] [nvarchar](50) NOT NULL,
	[DateWhenDue] [datetime] NULL,
	[DateCompleted] [datetime] NULL,
	[HoursOverDue] [int] NULL,
	[MaintenanceCost] [float] NULL,
	[PlannedOrUnplanned] [nvarchar](50) NULL,
	[DateStaffWasAssignedWork] [datetime] NULL,
	[HoursStaffOverDue] [int] NULL,
	[AssetName] [nvarchar](50) NULL,
	[ScheduleID] [nvarchar](50) NULL,
	[RequestID] [int] NULL,
 CONSTRAINT [PK_MaintenanceHistory] PRIMARY KEY CLUSTERED
(
	[HistoryID] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
END
GO
/****** Object:  Table [dbo].[StoresTable]    Script Date: 08/10/2009
15:05:46 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[StoresTable]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[StoresTable](
	[ItemCode] [nvarchar](50) NOT NULL,
	[ItemName] [nvarchar](50) NULL,
	[CategoryCode] [nvarchar](50) NULL,
	[Location] [nvarchar](50) NULL,
	[QuantityRemaining] [int] NULL CONSTRAINT
[DF_StoresTable_QuantiyRemaining]  DEFAULT ((0)),
	[RestockLevel] [int] NULL CONSTRAINT
[DF_StoresTable_RestockLevel]  DEFAULT ((0)),
	[DateOfLastRestock] [datetime] NULL CONSTRAINT
[DF_StoresTable_DateOfLastRestock]  DEFAULT (getdate()),
	[QuantityOfLastRestock] [int] NULL CONSTRAINT
[DF_StoresTable_QuantityOfLastRestock]  DEFAULT ((0)),
	[Description] [nvarchar](100) NULL,
	[AutoID] [int] IDENTITY(1,1) NOT NULL,
 CONSTRAINT [PK_StoresTable] PRIMARY KEY CLUSTERED
(
	[ItemCode] ASC
```

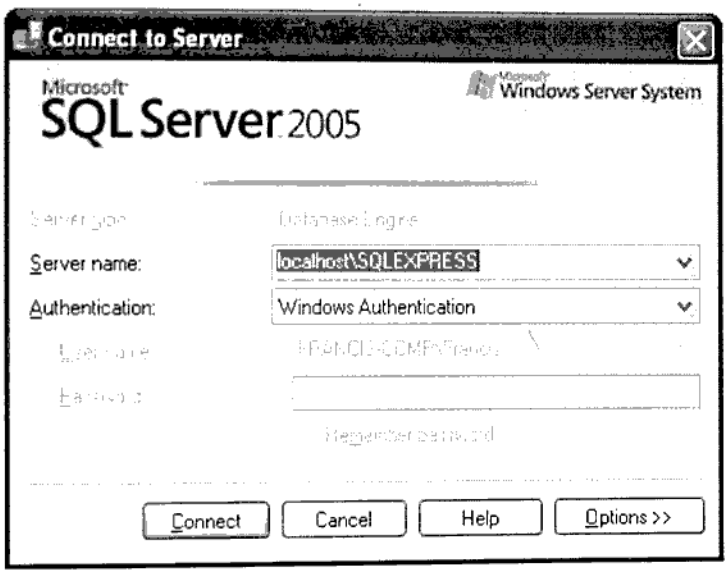
```
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF  ON [PRIMARY]
) ON [PRIMARY]
END
GO

/****** Object:  Index [IX_StoresTable]    Script Date: 08/10/2009
15:05:46 ******/
IF NOT EXISTS (SELECT * FROM sys.indexes WHERE object_id =
OBJECT_ID(N'[dbo].[StoresTable]') AND name = N'IX_StoresTable')
CREATE UNIQUE NONCLUSTERED INDEX [IX_StoresTable] ON
[dbo].[StoresTable]
(
      [ItemName] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF  ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Fault]    Script Date: 08/10/2009
15:05:46 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[Fault]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[Fault](
      [FaultID] [int] IDENTITY(1,1) NOT NULL,
      [FaultName] [nvarchar](50) NULL,
      [FaultDescription] [nvarchar](100) NULL,
 CONSTRAINT [PK_Fault] PRIMARY KEY CLUSTERED
(
      [FaultID] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
END
GO
```

Appendix C: Sample Code

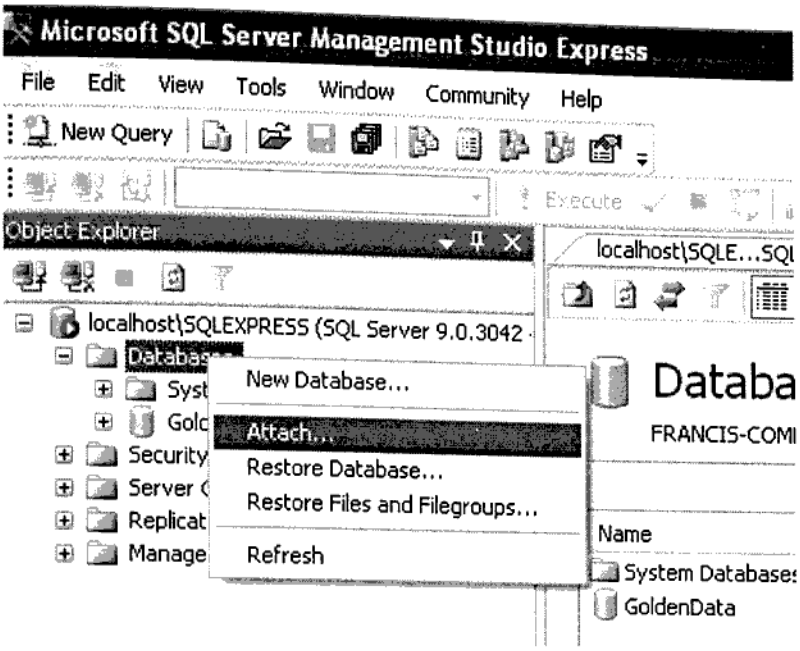
```
Public Function Update(ByVal updateString As String) As Boolean
        Dim querry As String
        Dim con As New SqlConnection()
        con.ConnectionString = Module1.Connection
        Try

            querry = updateString
            Dim cmd As New SqlCommand(querry, con)
            con.Open()
            cmd.ExecuteNonQuery()

            cmd.Dispose()
            con.Close()
            'success = True
            Return True
        Catch ex As SqlException
            If ex.Number = 2627 Or ex.Number = 2601 Then
                MessageBox.Show("The record you are trying to add
already exists", "An Error has occured", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            Else
                MessageBox.Show(ex.Message + ex.Number.ToString(), "An
Error has occured", MessageBoxButtons.OK, MessageBoxIcon.Error)
            End If
            If con.State = ConnectionState.Open Then
                con.Close()
            End If
            Return False
        End Try


    End Function
```

Appendix D: Installation of GTMMS

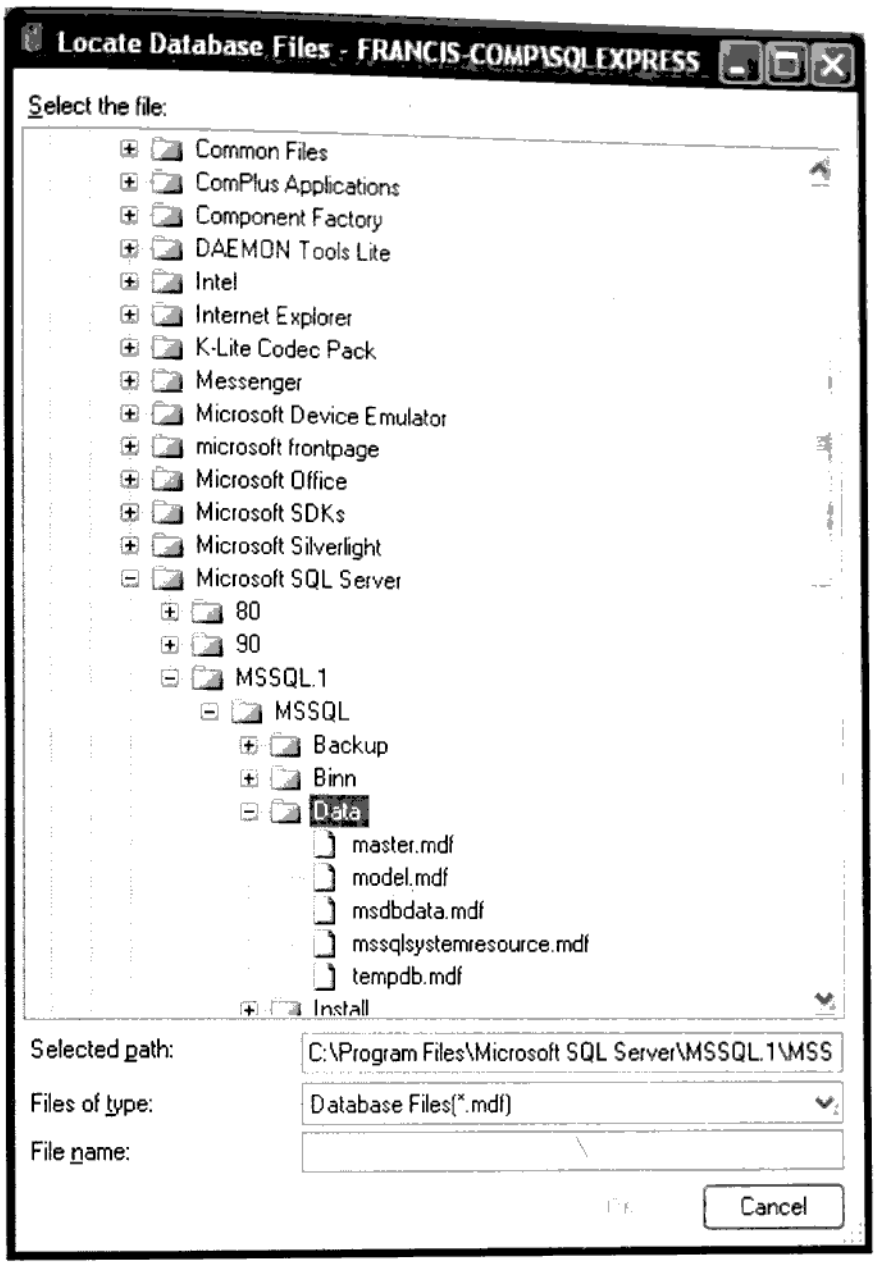
1. Locate Microsoft SQL Server Express Edition (SQLServer2005_SSMSEE.msi) from the Software\MManager\sqlexpress folder on the installation CD-ROM
2. Install Microsoft SQL Server Express Edition using the default settings
3. Locate Microsoft SQL Server Management Studio Express (SQLServer2005_SSMSEE.msi) from the Data folder on the installation CD-ROM
4. Install Microsoft SQL Server Management Studio Express edition
5. Start Microsoft SQL Server Management Studio Express



6. Click Connect button

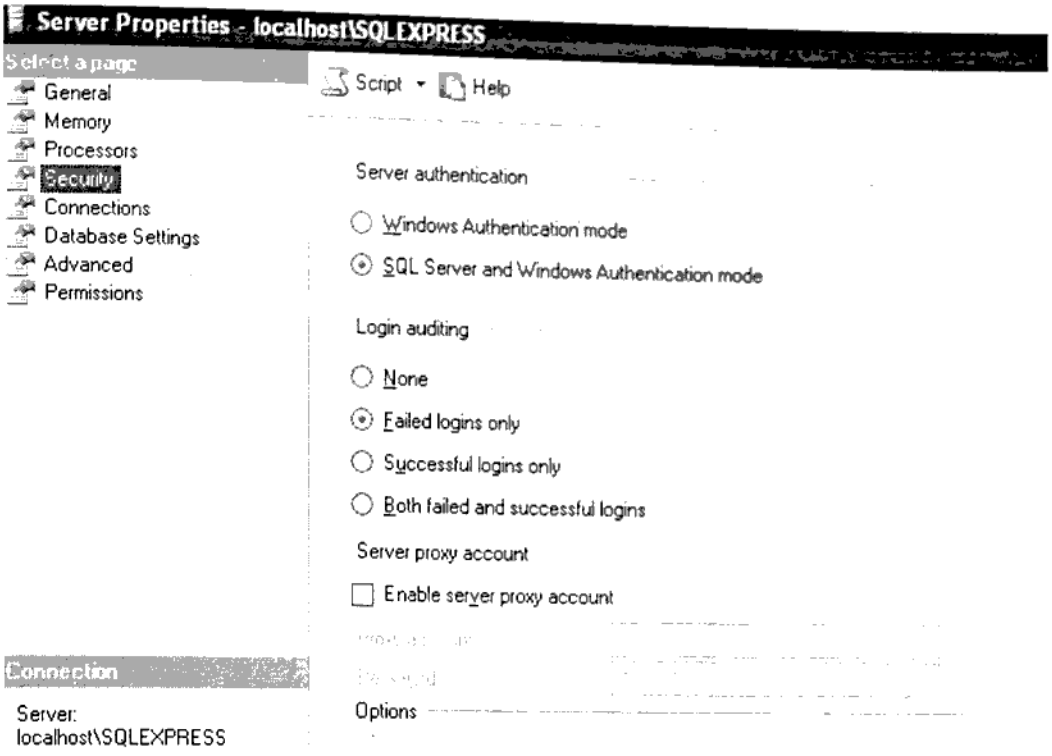


7. Right click the Database Folder and Select attach



8. Navigate to the location of the database and click OK

10. Right click the server icon and select properties



11. Select the Security option

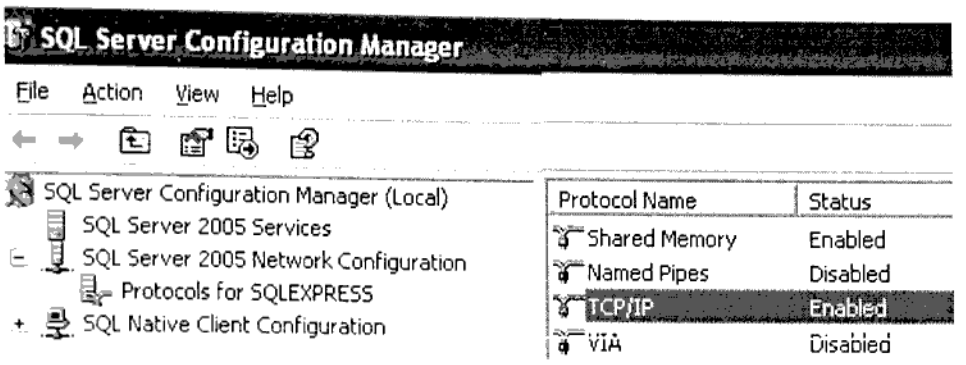
12. Under "Server Authentication'" select "SQL Server and Windows Authentication"

13. Click Ok

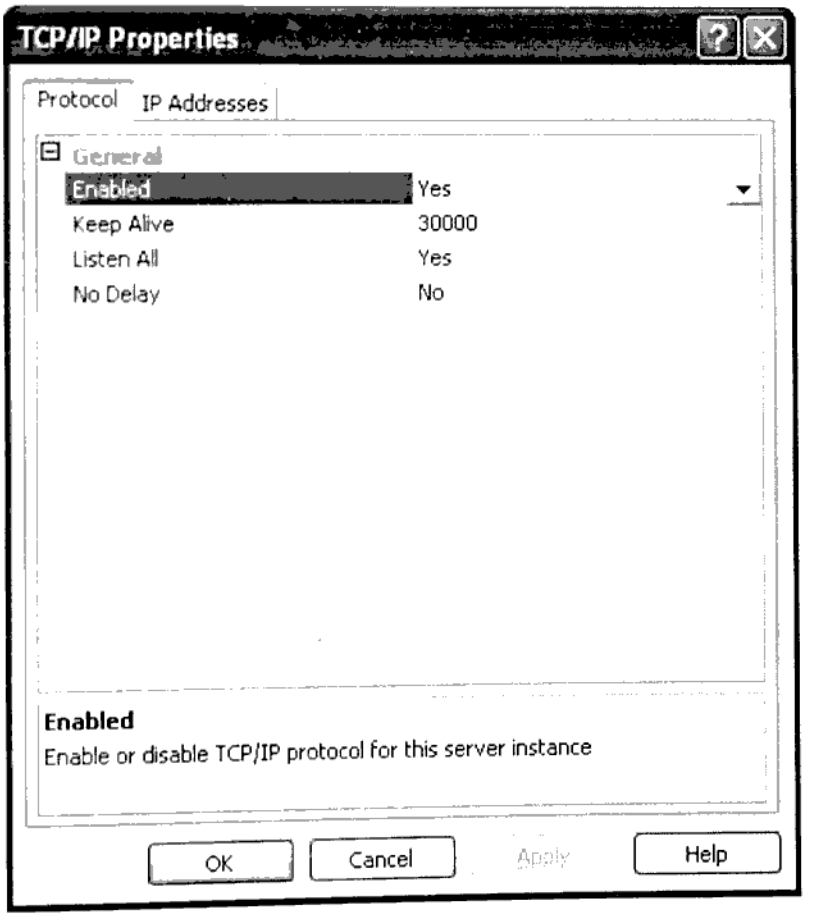
14. Navigate through the Windows Start Icon > All Programs > Microsoft SQL Server 2005 and select SQL Server Configuration Manager

15. Select the SQL Server Network Configuration Node > Protocols for SQLExpress

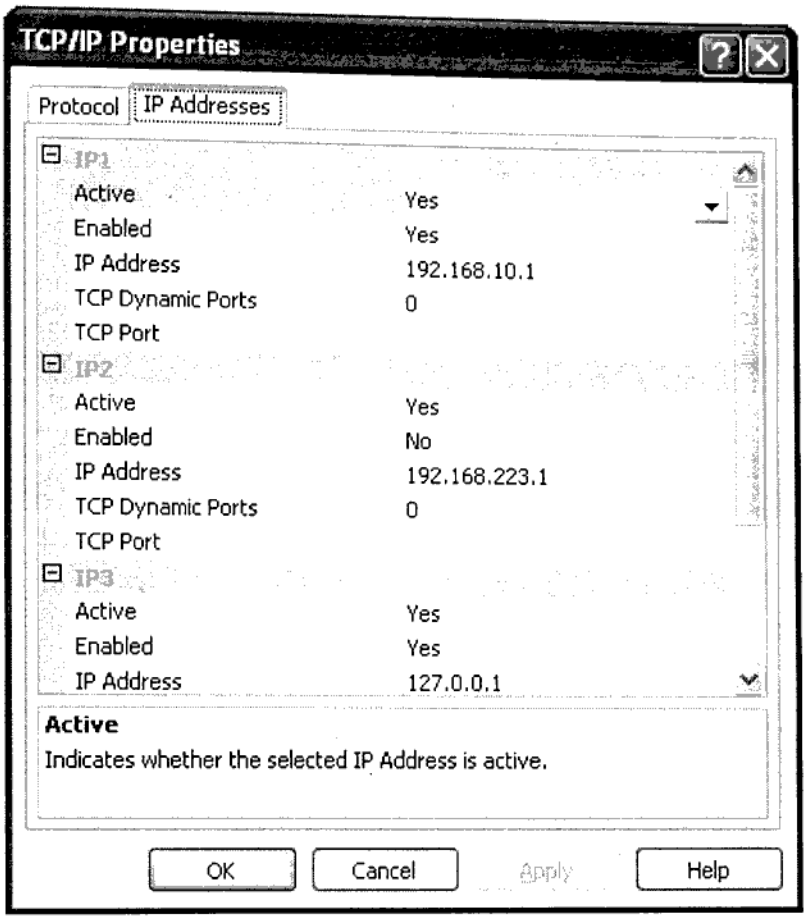
16. Right Click the TCP/IP option and select properties



17. On the Protocol tab, set the enabled option to Yes



18. On the IP Addresses tab, enable the IP address of the network interface card of your computer and set the 'Active' status to 'Yes'



19. Click OK and restart the server

20. Locate the LOGINGENERTOR.exe supplied on the CD from Software\Login\Debug and click the Generate Login button

21. Close the login generator.

22. Locate the setup.exe file supplied on the CD from Software\MManager and install using the default settings

23. For each client that needs to be installed , Locate the setup.exe file supplied on the CD from Software\Clientand install using the default settings

24. Start the GTMMS Maintenance Manager.

25. Use default username 'admin' and password 'gold' to login

Appendix D: GTMMS Usage Manual

For the manual on the usage of the GTMMS, locate the GettingStarted.html file from the Manual folder on the installation CD-ROM

OSEI TUTU II INSTITUTE FOR ADVANCED ICT STUDIES
GHANA

AFFILIATED TO

KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY, KUMASI

SCHOOL OF GRADUATE STUDIES

THE DEVELOPMENT OF SOFTWARE FOR MAINTENANCE MANAGEMENT IN THE HOSPITALITY INDUSTRY

(Case Study Golden Tulip Kumasi City Hotel)

A DISSERTATION PRESENTED TO THE OSEI TUTU II INSTITUTE FOR ADVANCED ICT STUDIES IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE AWARD OF THE DEGREE OF
MASTER OF SCIENCE IN ADVANCED ICT STUDIES
(MANAGEMENT OPTION)

BY
STEPHEN KWADWO OSEI
OCTOBER, 2009

DECLARATION

I hereby declare that this study was under taken independently and it is my original work.

It is not replication of any work either published or unpublished. All references made in this study are dully acknowledged. Finally, all aspects of this study have been discussed with and approved by my supervisor, Prof Vasko Fournadjiev

Signature ..... Date 21/10/2009

STEPHEN KWADWO OSEI
(STUDENT)

I declare that this dissertation was written under my supervision and that the Student has been consistent in his interaction with me for guidance and direction. He has my consent to present it for assessment.

Signature ..... Date .....

PROF. VASKO FORNADJIEV

(SUPERVISOR)

DEDICATION

This work is dedicated to Mr. Osei Frimpong, Ms. Agnes Boateng, Yvonne, Helen and Madam Cecilia Duku.

To MJ.

ACKNOWLEDGEMENTS

I express my sincere gratitude to Prof. Vasko Fournadjiev who has been a great mentor through the two intense years spent at the Institute and who whole-heartedly supervised my project work. God bless you richly for your immense help.

Many thanks to Prof. Martin Looijen, the rector of the Institute, for his guidance, and the entire staff for their support.

To Mrs. Sophia Quashie-Sam for understanding our plight and standing by us the entire time.

To Harmien Dam, whose unique sense of humour cheered us up when the going got tough.

To the General Manger and staff of the Golden Tulip Kumasi City hotel for their warm reception during my internship.

I would also like to thank Mr. Emmanuel Afful , Godfred Adzokatse and MT for their immense assistance and encouragement. Emma, thanks a million!

Special thanks to Mr. Osei Frimpong, Agnes Boateng and my grandmother, Madam Cecilia Duku for their love and support through the years. To my siblings Yvonne and Helen, I say, thanks a lot.

To the Ultimate Group, for coming up with the ultimate ideas and putting us on top.

To God Almighty.

STEPHEN K. OSEI

OCTOBER 2009

ABSTRACT

The objective of this study was to develop a maintenance management information system for the hospitality industry. The Golden Tulip Kumasi City Hotel was used as a case study.

The software consists of an asset classification module which places assets into a tree structure for easy identification. Along with this module is a room number generator and locator for the assets.

The software also catalogues a list of staff capable of maintenance services together with their working shift time table.

At the core of the software are the planned maintenance module, which has the responsibility of automatically processing preventative maintenance routines, and the unplanned maintenance module with which staff can report of assets that need maintenance outside regular schedules. The software then suggests a list of capable staff from which the maintenance administrator can choose from. If desired, the software is also able to automatically dispatch work orders to staff based on fair distribution of work load.

The final module tracks the history of the assets, which enables management to make informed decisions on the assets.

The project uses client-server architecture and is designed to work in a computer network environment.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CMMS: Computerised Maintenance Management System

SQL: Structured Query Language

DBMS: Database Management System

RDBMS: Relational Database Management System

GUI: Graphical User Interface

RAD: Rapid Application Development

OLEDB: Object Linking and Embedding Database

TCP/IP: Transmission Control Protocol /Internet Protocol

PMU: Planned Maintenance Unit

MM: Maintenance Manager

SUV: Sports Utility Vehicle

VW: Volkswagen

ID: Identity

PDF: Portable Document File

1 INTRODUCTION

1.1 Background

Every organization has assets it employs for its day to day activities. To stay in good running condition and to prevent incurring heavy cost of repairs, some of these assets need to be maintained periodically. The hospitality industry forms a group of such organizations that requires regular maintenance of assets to be able to provide top class services with minimum interruptions.

1.2 Problem Definition

The Golden Tulip Kumasi City Hotel is a relatively new hotel that was established in January 2008. The hotel has made heavy investments in a lot of equipment. Most of these equipment need to be serviced periodically. There are also lots of requests for maintenance that are not planned. For these reasons, it has become extremely difficult for the hotel to track down all maintenance activities and to attend to them on time. There is also the need to track the associated cost of maintenance and to isolate problematic equipment categories for the hotel's management to make informed cost decisions.

For these reasons, this thesis aims at developing software to manage the maintenance activities.

1.3 Objective of Study

The objective of this study is to develop an information system for maintenance management for the hospitality industry.

The system should have the following functionalities:

- Categorizing assets for easy identification,
- Cataloging a list of staff capable of maintenance services together with their working shift time table,
- Generating automatic maintenance schedules,
- Receiving request for unplanned maintenance from remote ends,

1.4 Justification

This thesis aims at developing an information system for managing maintenance activities in the hospitality industry. The Golden Tulip Kumasi City Hotel was taken as a case study. This was in view of the fact that there was no computerized maintenance management system in place. Request and scheduling activities in the maintenance department which could be made easier with a computerized information system were done manually and thereby were prone to human errors and slow response times.

The benefits of an information system for maintenance management include but not limited to:

- Reduction of equipment downtime by the ability to promptly report faults.
- Minimization of the frequency of equipment breakdowns by enhancing preventative maintenance scheduling.

- Reduction of maintenance cost due to minimized equipment breakdowns.

1.5 Methodology Used

The methodology used comprises:

Literature Review

- This part involves the review of literature on maintenance management
- It also involves finding relevant articles, technical documentation and papers on approaches for developing software applications.

Interviews

- Interviews were conducted with the hotel staff for their requirements.

Requirement Analysis and Software design encompasses:

- Understanding what is required of the system
- Consideration of which software development architecture to employ
- Choice of programming environment and database
- Program flow
- Database design

Software Coding

- This stage involves the writing of software code.

Testing and Evaluation

- Sample case studies are fed into the system to ascertain if the correct functionality and results are obtained
- Debugging will also be done at this stage.

1.6 Scope

This thesis focuses on the technical development aspect and does not extend to the soft managerial issues.

1.7 Limitations

The staff Work-Shift (time table) assumes a single shift per staff per day and does not cover multiples shifts per staff in a day.

The Work-Plan is only drawn for planned schedules since problems in unplanned maintenance activities may not be known in advance.

1.8 Thesis Outline

Chapter 1 provides an introduction and overview of this document. Chapter 2 delves into maintenance management and associated software used in maintenance activities as well as the review of some of the tools that are used for the design of the software. Chapter 3 gives the maintenance processes in existence at the hotel and tries to determine which ones are automatable and which ones are not. Chapter 4 discusses the architecture used in the thesis. Chapter 5 deals with the database and software components of the system. Chapter 6 deals with the design and implementation of the system. Finally, Chapter 7 concludes the thesis.

2 LITERATURE REVIEW

2.1 Management

Management can be defined as the art of handling or controlling something successfully or the skilful handling or use of resources. (Encarta, 2009). It entails planning, allocation of resources to the plan and evaluation to see if the end result(s) is/are in line with what was expected (Shead, 2007). It can be inferred that the key element in management is coordinating all components within a given system for a desired outcome.

A set of processes and procedures that an institution or organisation uses to achieve such an outcome forms what is known as a management system (Wikipedia, 2009).

2.2 Maintenance

Maintenance can be defined as actions that are required to keep a machine or system in service (Wiktionary, 2009). This could include but not limited to, replacement of worn out parts, lubrication, cleaning, as well as other servicing performed on assets.

Putting these two definitions together, we could propose that Maintenance Management is the skilful handling of the actions that are required to keep a machine or system in service.

2.3 Maintenance Management System

Maintenance management deals with "the process of leadership, organizing, planning and scheduling" of preventative maintenance, condition monitoring, and execution of maintenance repairs and recording of technical data in support of management (IDCON, 2009). A Maintenance Management system is, therefore, one that is designed in support of the above. When such a system is aided by computer software, it is known as Computerised Maintenance Management System (CMMS).

The CMMS stores an information base of the maintenance activities undertaken by an organisation. The stored information helps in classification and easy location of asset for easy identification and application of maintenance schedules as well as helping management take reasonably informed decisions (Wikipedia, 2009).

CMMS may be specific to the maintenance operations of an organisation or can be generic serving the needs of several organisational types. For example, the needs of a health care organisation may be different from that of the hospitality industry. They could also have some routines in common. CMMS may include one or more of the following (Wikipedia, 2009):

- Management of Asset: Consists of keeping records of assets. Such records may contain information pertaining to warranty, service contracts, life expectancy of asset, dates of purchase. Generally, it keeps information that may be relevant to management and maintenance work.
- Orders for Work: It comprises but not limited to, scheduling of jobs, dispatching of staff to the job, and documenting the cost of repairs/service.

- Inventory Management: Includes management of materials, equipment and replacement parts, determining restock level and order placement for new stock.

2.4 Review of Some Software Systems for Maintenance Management

2.4.1 Bigfoot CMMS

Bigfoot CMMS is designed by Smartware Group (Smartware Group, 2008). It is a multi-sector computerised maintenance management system designed for the healthcare, hospitality, education, financial, food and manufacturing industries.

Bigfoot comes in two versions: The 'Traditional' version and the 'Internet' version.

The 'Traditional' version is ideal for small facilities where the software database is installed on a local server and access to the internet is minimal or nonexistent. It is faster than the 'Internet' version.

The 'Internet' version is ideal for corporations with several facilities whose maintenance activities need to be managed. This version provides access to the system from virtually any place where there is internet availability (Smartware Group, 2008).

Bigfoot presents a generic solution for maintenance management and does not focus specifically on the hospitality industry.

2.4.2 CWorks CMMS

CWorks maintenance management system is designed by CWorks Systems Berhad. It is a CMMS for managing the maintenance activities of medium sized organisation. It has an asset module for cataloguing assets, a location module for locating asset, work request module for receiving work requests, a work scheduling module, employee

information module and a reports module for generating reports. CWorks is built on a Microsoft access database and as such not too well suited for enterprise-wide use (CWORKS SYSTEMS, 2009). It lacks support for managing the time-tables of employees.

2.4.3 Clever CMMS

Clever CMMS is another CMMS with support for maintenance scheduling, asset management, report generation. It is designed by Clever Solutions. It has a user friendly interface and supports multiple concurrent users. It also has an inbuilt messaging system for intercommunications between users (Clever Solutions, 2009).

Clever CMMS is also a generic system that does not specialize in the specific needs of the hospitality industry.

2.5 Client-Server Architecture

A server can be defined as an application that provides a service that is requested for by another application. A server generally sits and waits for a request before responding to that request. A client on the other hand, request for a service from a server. It is the consumer of the service (Ahmed, Garrett, & Faircloth, 2002)

A Client-Server Architecture separates the presentation layer (user interface) and data access layer (database) by running them logically or physically separate platforms. The business logic may be kept on either of the two platforms or can be on its own platform.

Client-Server applications can also be referred to as multi-tier or n-tier applications. The simplest of multi-tier applications is the two tier application. Figure 2.1 shows a diagrammatic representation of a simple two-tier application. Some of the advantages of a multi-tier application include (Sempf, 2005):

- Replacing one component does not require changing the rest (modular design)
- Replacing one component does not affect the functionality of the rest (modular design)
- Placing each component separately enhances:
    - Security: Since access to one component does not guarantee access to the others.
    - Reduced Load: If a client is involved in some processing, such load is taken off the server.
    - Availability: As a result of the reduced load where each client is involved in some processing, there is minimal congestion and this contributes to the availability of the service.(An exception is when the server does all the processing and pushes the result out to the client)

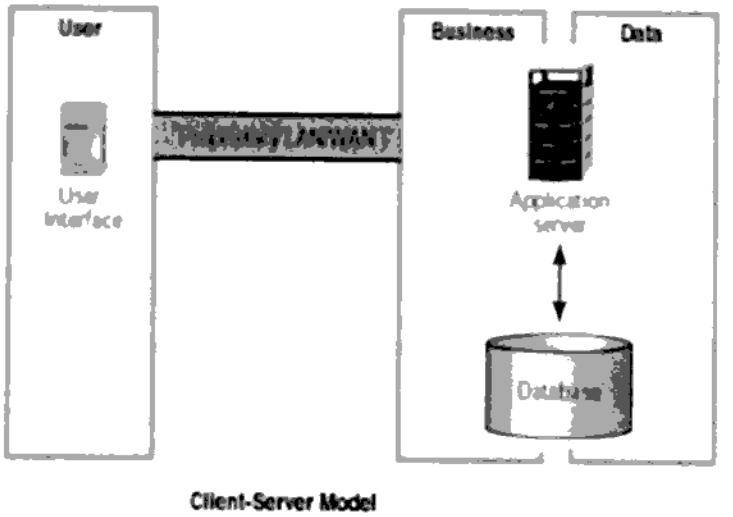


Figure 2.1 Two-tier client server model

The stated advantages above makes client-server architecture well suited for the development of a maintenance management system. Such a maintenance management system will take requests from staff at different sites for the review of the maintenance administrator.

2.6 Relational Databases and Relational Database Management System

A database can be defined as a structure for storing data which is arranged for efficient and fast retrieval (Petroutsos & Bilgin, 2002). A database refers to a structured organised group of records (Price, 2003).It is a repository of information (McManus & Goldstein, 2003). The data they contain can be queried in many ways for information retrieval. Databases are essential for a multi-tier (client-server) system (Bryla, 2003). They provide the backend to maintenance management applications and indeed, most information systems related applications. A relational database is a database which groups related data into tables and defines logical links (relationships) between the tables (Kauffman, Claudio, Chinnathampi (Das), Greenwood, & Matsik, 2002). It uses two-dimensional tables (relations) to store information for supporting a business (Bryla, 2003).Figure 2.2 shows two tables with a relationship defined.

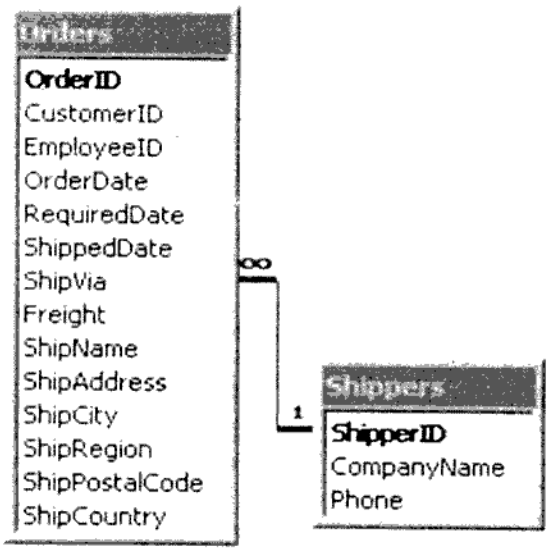


Figure 2.2 Tables with a relationship defined

Advantages of using a relational database include (Buzzle.com, 2008):

- Support of Normalization: Normalization refers to the step by step process of removing redundancies(repetitions) from a database and ensures that the database is free from undesirable characteristic relating to the update, deletion or addition of data (Wkipedia, 2009)
- Support of Structured Query Language (SQL): SQL a database language that is easy to understand and with a syntax that is human readable. SQL is also supported by many vendors
- Support for Dynamic Views: A view is a snapshot of data that can comprise data from several tables. A relational database view is not static and can change on-the-fly as the underlying data changes. Views are used to simplify complex data by presenting only the entries that are needed.
- Support for Security: Relational databases provide the functionality of limiting access rights.
- Relational databases are scalable and well suited for the implementation of a Multi-tier(client-server) application.

A Database Management System (DBMS) controls the creation, utilization and maintenance of databases. A DBMS designed to work on relational databases is known as Relational Database Management System (RDBMS) (Price, 2003).

2.7 Software Development Process

Software development process refers to a guideline structure that is undertaken for the development of software (Wikipedia, 2009) .

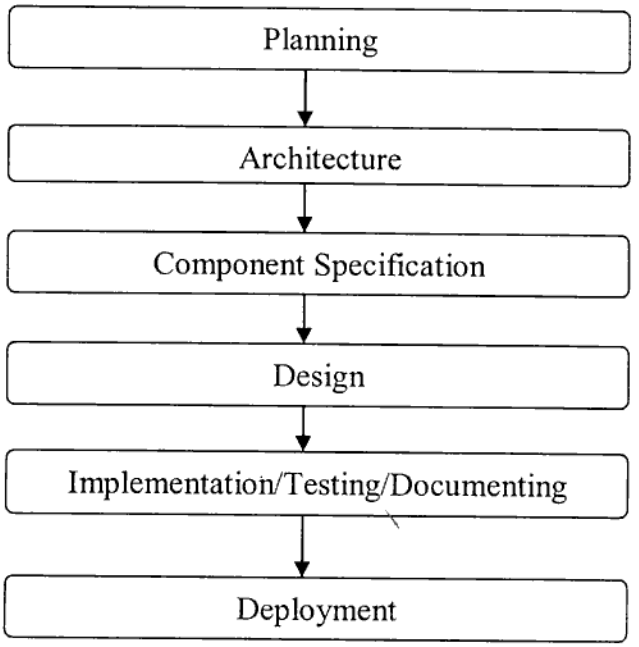


Figure 2.3 Software Development Process

The activities involved in the Software Development Process, also known as the Software Life Cycle are (Wikipedia, 2009) :

- Planning: This involves the gathering of requirements from end users and analysing those requirements. Scope of the project is defined at this point.
- Architecture: This refers to an abstract representation of the system. It aims at ensuring that the software will meet its laid down requirements.
- Component Specification: This involves a precise description of the software components and functionalities to be written

- Design: This stage encompasses the logical design of both the database and interfaces used in the system.
- Implementation, Testing and Documenting:
    - Writing of software code encompasses the implementation stage.
    - Testing is done to identify bugs for correction
    - Documentation of the design of the software is crucial for the future maintenance of the software.

2.8 Software Programming Languages

A programming language is a form of non-human language designed to enable machines perform specific tasks. As pertains to computers, a programming language can be defined as a language used to instruct computers to perform a task of computation or algorithm as well as controlling other devices. (Wikipedia.org, 2009)

2.8.1 CodeGear Delphi

This programming language is developed by CodeGear. It has its roots in the Pascal programming language. Delphi is well suited for desktop and enterprise database applications. It is relatively easy to use and stable (Wikipedia, 2009).

2.8.2 Visual Basic 2005

Visual Basic 2005 is developed my Microsoft. It has its roots in the BASIC programming language. It is a Rapid Application Development (RAD) tool and has rich Graphical User Interface controls (GUI). The most distinguishing feature of all BASIC programming languages is that they are easy to read and understand and follow an

English-like syntax. Visual Basic is robust and stable. The Express Edition is also available for free (Wikipedia, 2009).

2.9 Database Management Systems:

2.9.1 Microsoft Access

Access is an RDBMS developed by Microsoft and is part of the Microsoft Office application package. It is based on the Microsoft Jet OLEDB engine. Microsoft Access was developed for the desktop environment. It is relatively easy to install and administer. It is also cheaper than most RDBMS software. Access is not well suited for a Client-Server environment. It supports a limited number of concurrent users and does not scale very well as the amount of data grows to enterprise-level sizes (Dewson, 2006).

2.9.2 Oracle

This is a relational database management system developed by Oracle Corporation. This RDBMS has a high user base and is used extensively by search engines. Oracle is a very powerful and stable. It is highly scalable and fast. It offers flexibility by providing many add in tools. It is well suited for the Client-Server environment. Oracle is also noted to be quite complex to administer and expensive (Dewson, 2006).

2.9.3 Microsoft SQL Server 2005

SQL Server 2005 is developed by Microsoft Corporation. This RDBMS is stable, good for simple applications as well as scales well with large (enterprise) applications. SQL Server 2005 is relatively inexpensive. The Express edition is available for free. It is user-friendly and provides many tools for analyzing data. It is also relatively easy to install

and administer. The Express edition does not come with many of the analysis tools as the commercial versions and is not intended for very large commercial applications (Dewson, 2006).

2.9.4 Sybase

Sybase is a very fast and robust relational database management system. It is known to be rebooted, on average, only once or twice in a year. Sybase is predominantly used in the Unix operating system environment. It is administered mainly by command-line and is not very user friendly. Sybase has very limited Graphical User Interface (GUI) (Dewson, 2006).

3 PLANNING

3.1 Overview of Existing Maintenance Processes

The preliminary work on specification gathering for the software was done on site by observing the flow of work at the maintenance department of the Golden Tulip hotel. These processes are outlined in Figure 3.1

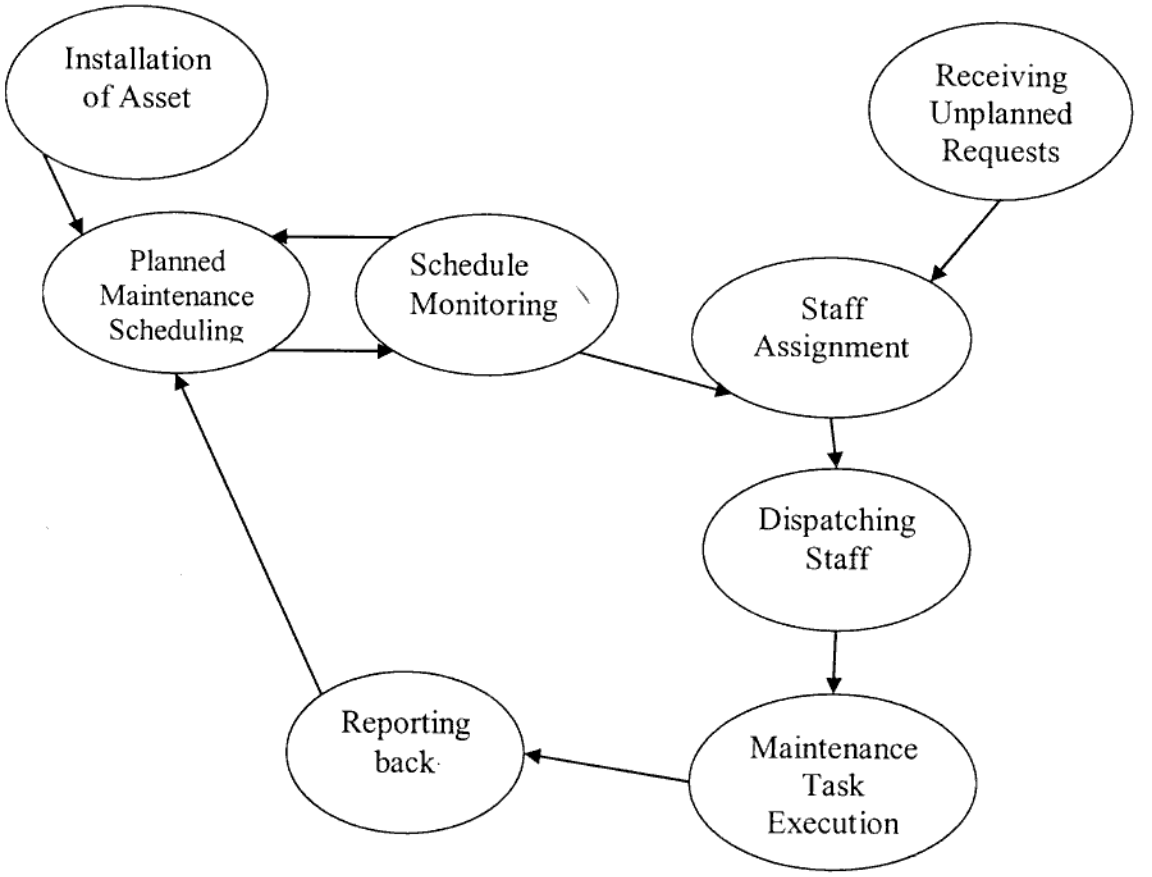


Figure 3.1 Outline of Existing Maintenance Processes

3.1.1 Installation of Assets

Actors: *Technical Staff, Maintenance Administrator.*

Input*: asset data, location data*

Output: *asset catalogue*

Typical steps for a new installation are as follows:

- A new asset (equipment) is catalogued and given an asset identification number (Asset ID). The catalogued information includes that specified by the manufacturer for its preventative maintenance.
- A technician with the requisite skills for installation is assigned and the equipment is taken to its designated location for installation.

3.1.2 Planned Maintenance Scheduling

Actors: *Maintenance Administrator*

Input: *Asset data*

Output: *maintenance schedule*

The following are the steps involved in creating a preventative (planned) maintenance schedules:

- Information from the manufacturer on the preventative maintenance interval is recorded in the cataloguing phase of the installation of the asset.
- If such information does not exist, the maintenance engineer/administrator together with a technician may provide such information based on experience with such equipment or based on discretion.
- A list of the recommended preventative maintenance information is created and printed out.

3.1.3 Schedule Monitoring

Actors: *Maintenance Administrator, Technical Staff*

Input: *maintenance schedule*

Output: *monitor schedule*

Monitoring of the planned maintenance schedule entails the following:

- Technical staff are given a copy of the schedule list and assigned to some of the equipment.
- The technical staff are supposed to go through the list on a daily basis and attend to the eequipment that need servicing.
- The maintenance engineer/administrator goes through the list and prompts the technical staff on their schedule.

3.1.4 Staff Assignment

Actors: *Maintenance Administrator*

Input: *staff skill data, service skill data, staff work load*

Output: *staff assignment*

Technical staff are assigned to schedules or tasks via the following procedure:

- Information on the technical capabilities of the staff is stored in a file
- The information on the asset that is due for maintenance is checked for the kind of skill that is required for the service.
- Based on skill required, a member of the technical staff is selected. This is also based on the work load on the staff.

3.1.5 Staff Dispatch

Actors: *Maintenance Administrator, Technical Staff*

Input: *maintenance schedule, staff assignment*

Output: *Staff Dispatch, Dispatch form*

- The maintenance administrator issues out a dispatch form to the staff assigned to the task. On the form are the location of the asset, the date of dispatch and the name of the asset.

3.1.6 Maintenance Task Execution

Actors: *Technical Staff, Maintenance Administrator/Supervisor*

Input: *Dispatch form, Staff Dispatch*

Output: *Physical Execution of task*

- The technical staff physically executes the task and reports back to the maintenance administrator
- The maintenance administrator/supervisor then inspects the work

3.1.7 Reporting Back

Actors: Maintenance Administrator, Technical Staff

Input: *Dispatch form, Schedule Data*

Output: *maintenance schedule*

- After inspection of the task, the dispatch list is signed by the maintenance administrator.
- The maintenance schedule of the asset is then updated.

3.1.8 Unplanned Maintenance Requests

Actors: *General staff, Guests*

Input: *Asset data*

Output: *Request Maintenance*

When a piece equipment breaks down, the owning department sends written or verbal request to the maintenance engineer/administrator. The administrator then assigns a technician with the requisite skills to the task. After execution, the technician reports back. The administrator then inspects the work. The administrator then informs the owning department of the outcome.

3.2 Automatable Processes

There exists the possibility of automating some of the maintenance processes or activities. Some processes can be fully automated and others partially. The under listed gives an outline of processes and activities that can be automated and those that cannot..

3.2.1 Installation of Assets

Level of Automation: *Partial*

The physical installation of an asset cannot be automated with maintenance management software. It requires human discretion and expertise. Installation procedure also varies from one asset to the other. However, cataloguing of the asset can be fully automated since it is primarily a record keeping activity. Selection of staff with the requisite skills can be automated since both the details of staff and the skill required can be electronically captured.

3.2.2 Planned Maintenance Scheduling

Level of Automation: *Partial*

The gathering of the initial information that is needed for scheduling cannot be automated. It requires human effort and discretion to gather this information, especially, when not stated by the manufacturer or the interval stated by the manufacturer does not yield acceptable results.

However, it is possible to feed the data gathered into a maintenance system and print out schedules electronically.

3.2.3 Schedule Monitoring

Level of Automation: *Full*

It is possible to fully automate the monitoring of maintenance schedules. It does not require human effort. It is possible for the software to prompt the maintenance administrator of an impending planned maintenance service.

3.2.4 Staff Assignment

Level of Automation: *Partial*

It is possible for software to assign staff to maintenance tasks. However, it requires a human being to judge the quality of work from staff to better make an assignment. The availability of the staff is best determined by a human. Based on this, the assignment of staff cannot be fully automated.

3.2.5 Staff Dispatch

Level of Automation: *Full*

Once a member of staff is assigned, software can be used to print the dispatch form with the details of the work to be done. This process can be fully automated.

3.2.6 Maintenance Task Execution

Level of Automation: *None*

The physical execution of a maintenance task cannot be performed by software. Human presence is also required for the inspection of the completed task. This process can, therefore, not be automated.

3.2.7 Reporting Back

Level of Automation: Partial

It is required of a supervisor to check on work that has been completed. The supervisor is also required to sign the dispatch form. The two activities cannot be automated. However, updating the maintenance status of the asset can be automated.

3.2.8 Unplanned Maintenance Requests

Level of Automation: *Partial*

It is possible to eliminate the manual transfer of unplanned maintenance request into an electronic form but it requires human presence to spot the fault and report it.

4 ARCHITECTURE

The GT Maintenance Management System (GTMMS) is the name of proposed system being developed. From the perspective of the end-user, it consists of three major components:

- The Unplanned Maintenance Client
- The Planned Maintenance Unit
- The Maintenance Manager

The Maintenance Manager manages the activities of the Planned Maintenance Unit and the Unplanned Maintenance Client. Figure 4.1 shows an overview of the GTMMS.

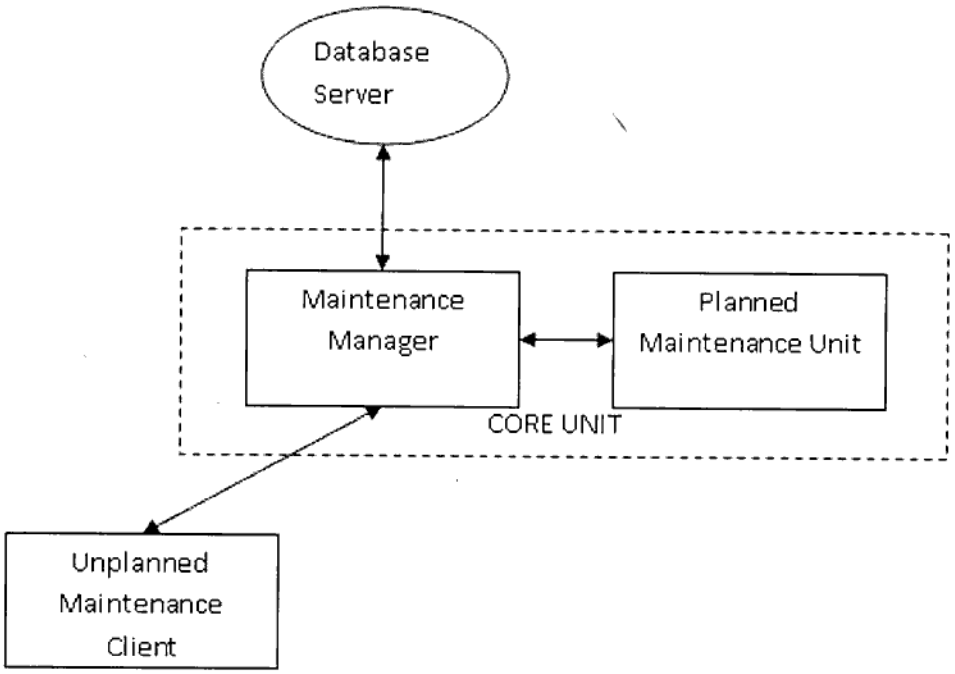


Figure 4.1 GTMMS Overview

The GT Maintenance Management System (GTMMS) has been developed on Client-Server paradigm. It is designed such that each of its segments can be run on different computers across a computer network using the TCP/IP Protocol.

It is segmented into three units:

- Unplanned Maintenance Client (User Requests and correspondence)
- The CORE Unit (Maintenance Manager, Planned Maintenance Unit)
- Database Server

All Cataloguing, Request and Scheduling data are first stored on the Database Server. The Maintenance Manager then retrieves them for analysis. The maintenance administrator/engineer then acts on the result of the analysis (Figure 4.2).

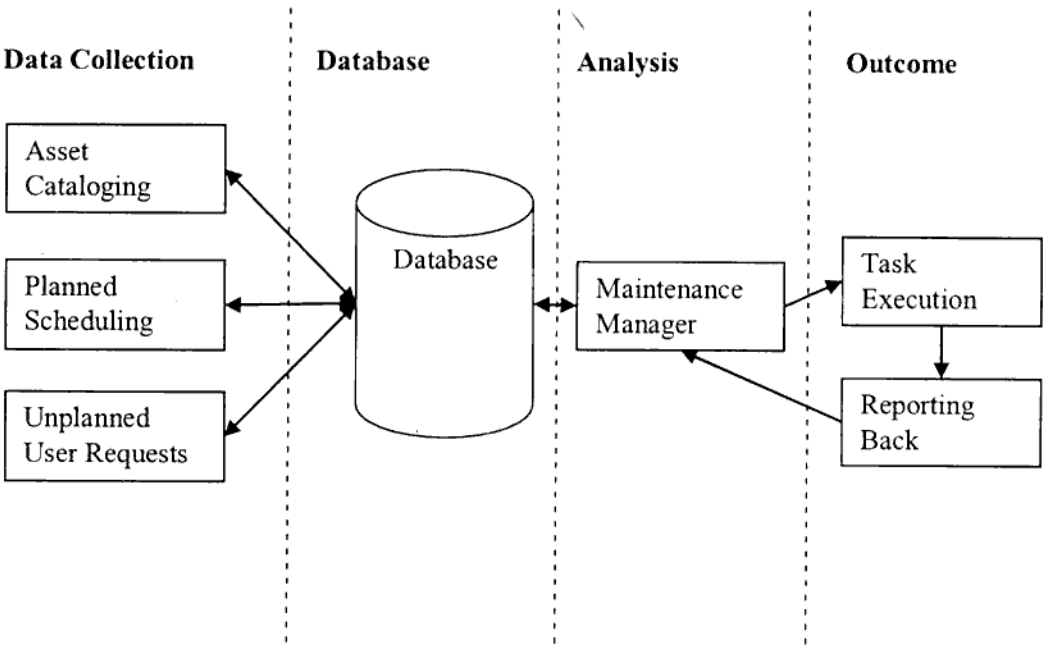


Figure 4.2 GTMMS Architecture

4.1 Unplanned Maintenance Client

The Unplanned Maintenance Client is responsible for taking user requests and sending them to the Maintenance Manager. It consists of a data table (figure 4.3) that houses the users' or departments' login identification details stored on the database server. There is also a data table (Figure 4.4) that stores the correspondences between the user and the maintenance administrator/engineer.

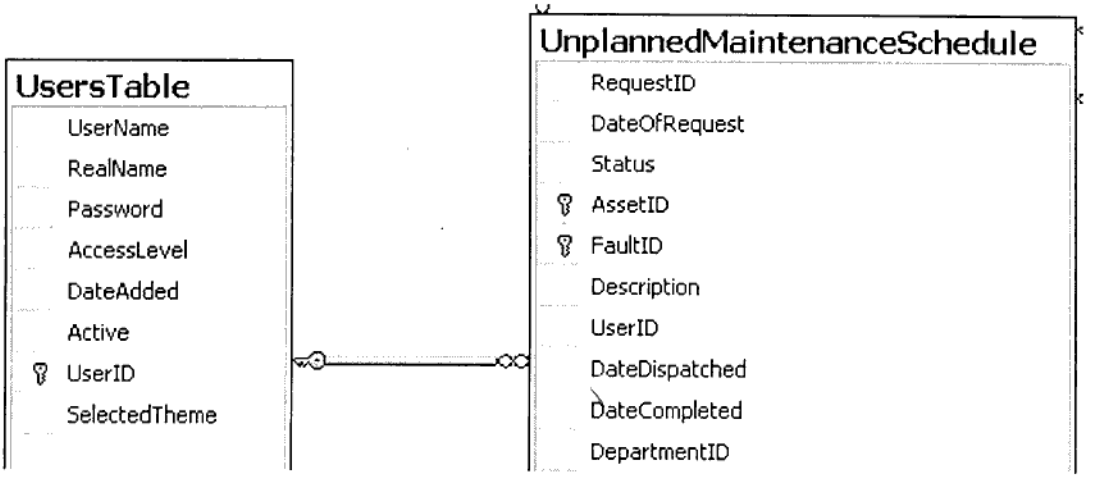


Figure 4.3 User Request Table

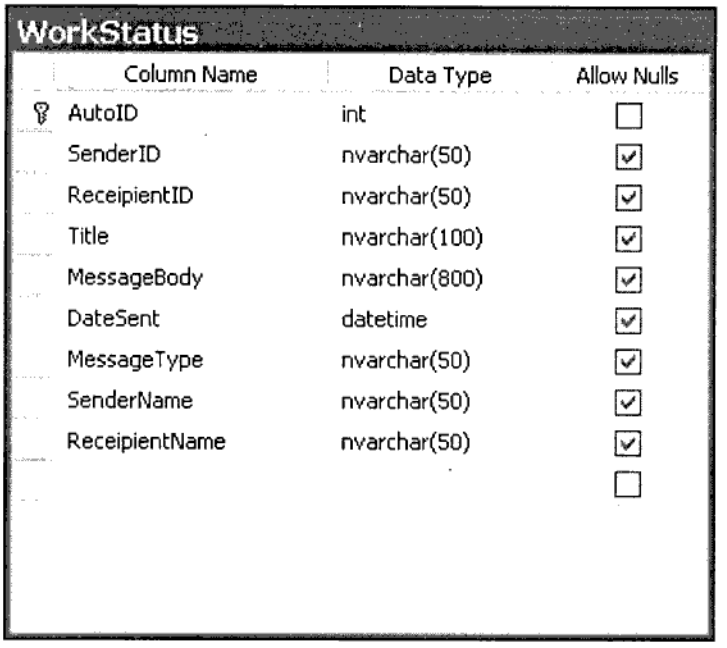


Figure 4.4 User Correspondence Table

Figure 4.5 is the activity diagram of unplanned maintenance.The user fills an electronic requisition form. The form has fields from which the user can select the location, the Asset ID number, the type of fault as well as give a brief description of the problem. The form then goes to the UnplannedMaintenance table. Upon receiving the request, the Core Unit's Maintenance Manager automatically notifies the user of the receipt. The Core Unit's Maintenance Manager then presents the request to the maintenance administrator. The Maintenance Manager also suggests a list of staff with the competence to handle the problem. The maintenance administrator then reviews the suggestions and makes the necessary changes and dispatches the staff to handle the job. The user is also informed of the dispatch via the Maintenance Manager.

Upon execution of the task, the maintenance administrator, after inspection, updates the job status. A report is sent to the user on the status.

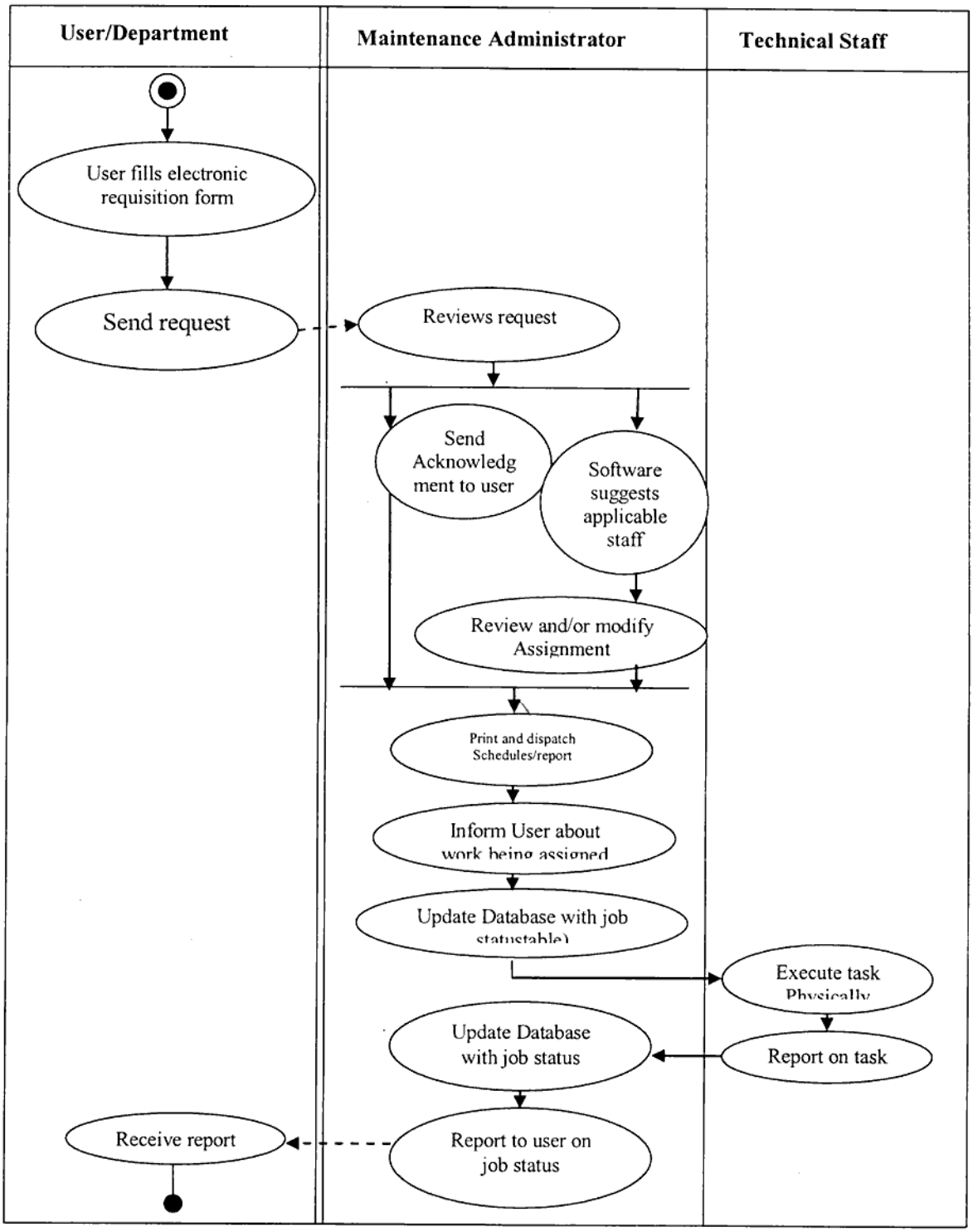


Figure 4.5 Activity Diagram of the Unplanned Maintenance Client

4.2 The Core Unit

The Core Unit is the engine of GTMMS. It is responsible for processing all the maintenance requests as well as scheduling maintenance activities.

The Core Unit has two sections based on their functions. These are:

- The Planned Maintenance Unit
- The Maintenance Manager

4.2.1 Planned Maintenance Unit

The Planned Maintenance Unit is responsible for drafting planned maintenance schedules of assets and monitoring those schedules. It provides an interface for assigning maintenance schedules to assets either individually or as applied to a whole class (group) of assets. The Planned Maintenance Unit integrates seamlessly with the Maintenance Manager for the execution of planned maintenance activities. Figures 4.6 and 4.7 shows the flow and activity diagrams of the Planned Maintenance Unit (PMU)

The following describe the operations of the Planned Maintenance Unit :

- The planned maintenance schedule of the asset is drafted via the PMU
- The PMU reads the schedule and calculates the due date for maintenance
- When a schedule is due, the Maintenance Manager suggests a list of staff with the capability of performing the task to the maintenance administrator.
- The maintenance administrator then selects the staff and assigns them the task. Optionally, the Maintenance Manager could automatically assign a staff based on a fair queue algorithm that the maintenance administrator can review and modify.

- Upon completion of the task, the maintenance administrator updates the status of the schedule.
- The next schedule date is then re-calculated by the PMU

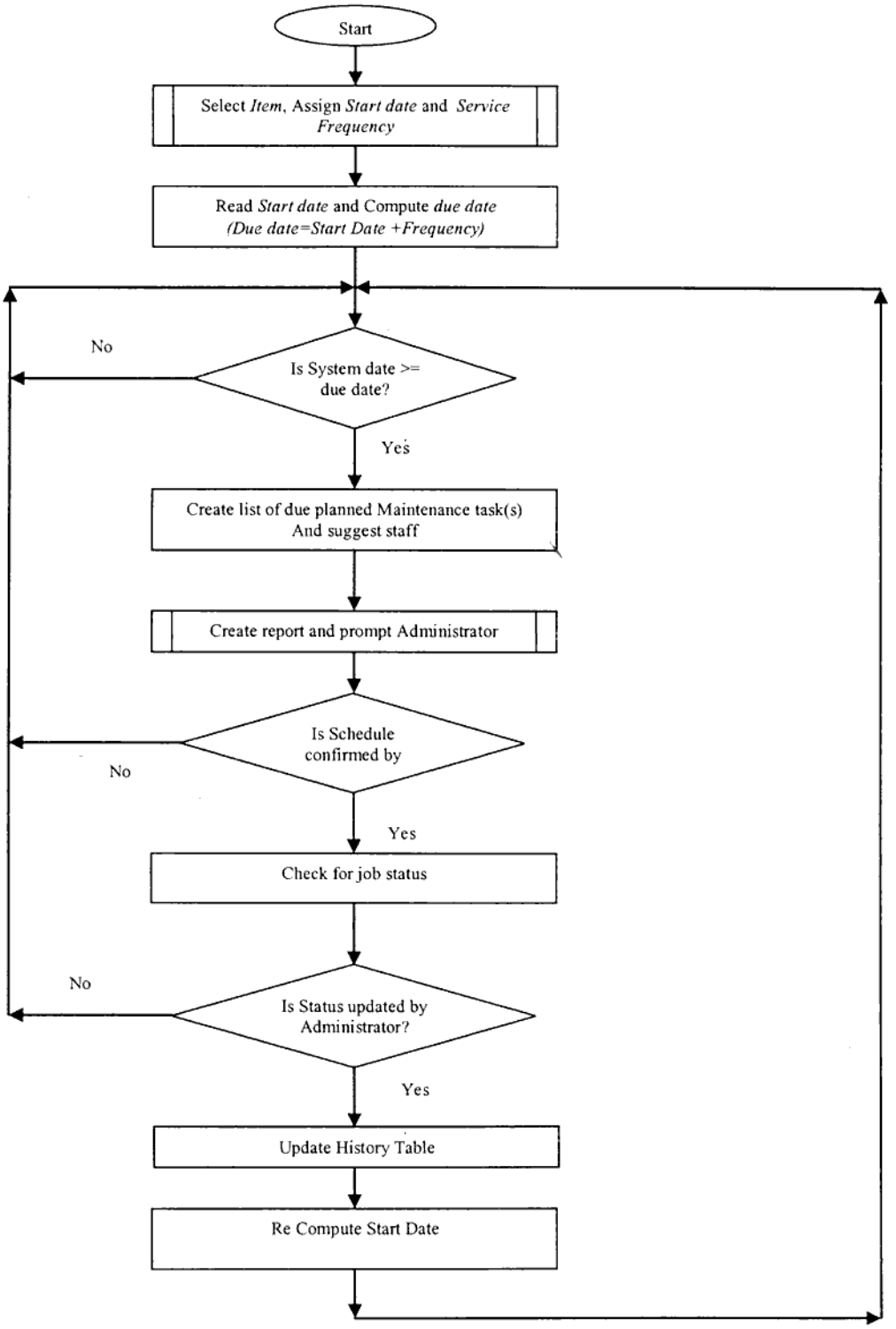


Figure 4.6 Planned Maintenance Flow Chart

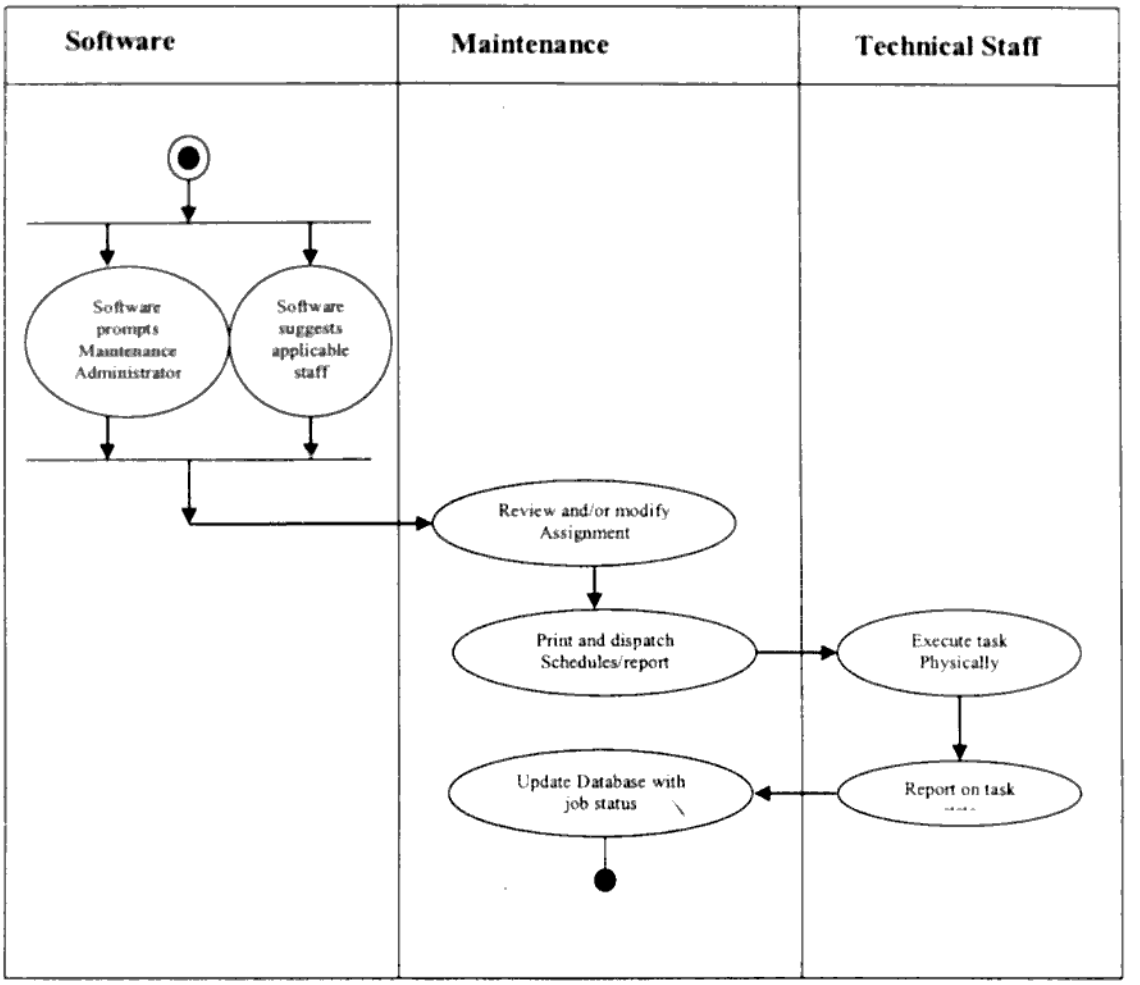


Figure 4.7 Planned Maintenance Activity Diagram

4.2.2 The Maintenance Manager

The Maintenance Manager (MM) is the bedrock of the GTMMS. It coordinates integrates all the other components of the software. It is responsible for the following:

- Cataloguing of Assets:

  The Maintenance Manager is responsible for classifying assets into segments such as, categories, types, classes and brands.

- Generating Room Numbers

  The MM implements a function for generating room numbers for the storage of assets. There is also a module for locating rooms.

- Managing Staff Work Shifts

  The Maintenance Manager implements a time table module that enables the maintenance administrator manage the work shifts of the technical staff.

- Managing Staff Competencies (skills)

  There is a module for managing the capabilities of technical staff.

- Managing Faults and Services

  The Services and Faults module presents a list of services required for planned maintenance activates as well as a check list of the various step to be taken when servicing.

- Managing User Accounts

  The Maintenance Manger manages the accounts of user. This involves the creation and modification of the account and the assignment of privileges.

The Maintenance Manager is also responsible for tasks such as acknowledging users about the status of their requests, suggesting applicable staff for a task as well as automatically assigning staff.

4.3 Database Server

The hub of all the data involved in the software resides on the database server. The server incorporates:

- Data tables for storing data for later retrieval
- Data Views for looking at the data through certain filters and conditions
- Functions for performing some simple computational tasks
- Login Accounts for restricting access to the data

5 SOFTWARE AND DATABASE COMPONENT SPECIFICATION

5.1 Software Components

After observing and interviewing the staff on their maintenance activities, the following requirements were realised:

- To eliminate the need for manual checks on preventative maintenance schedules,
- To enable departments and individuals forward maintenance requests directly to the system.
- To provide information that can be used for decision making

From the points listed above, the following components (functionalities) were crafted out:

5.1.1 Categorizing assets for easy identification

The asset needs to be placed in various categories and sub categories for differentiation and easy access. Information that uniquely identifies each asset needs to be created. The location of the assets should also be known at every instant.

5.1.2 Cataloguing Staff and their competencies

Information on staff and their areas of expertise is needed for the selection of the appropriate candidate for a maintenance task. Information of their working shifts is also needed for the maintenance administrator to better assign a task.

5.1.3 Generating automatic maintenance schedules

A key requirement to eliminating the need for manual checks on preventative maintenance schedules is to be able to generate an automatic maintenance schedule. This is based on monitoring the preventative maintenance intervals and selecting the assets that are due for servicing.

5.1.4 Managing Staff Information

Information on technical staff is required for the selection of the appropriate staff for a task assignment. This information consists of the Staff general information, time-table, skills and the staff job burden (number of tasks assigned or performed).

5.1.5 Tracking Maintenance History

Tracking of the history of maintenance is essential for determining problematic assets and the cost of maintaining assets. This component is very important in generating reports for decision making.

5.1.6 Receiving requests from remote ends

The software is required to provide the functionality of receiving requests from departments and individuals. This cuts down the need to physically move from one location to the other. This functionality reduces paper work and delays in reporting an incident. The requesting party should also be given progress reports of the status of the request at each stage.

5.1.7 System Actors and their Corresponding Functions

The system actors are the end users of the Maintenance Management System. They have varying amount of privileges and responsibilities assigned to them on the use of the software. Below are the primary users and their responsibilities

Software Administrator(s)

- Installs database management software for data management
- Configures database to communicate with software application
- Installs software application
- Creates user account(s) and sets user privilege(s)
- Performs update or delete or add functions. (-basically, all tables)
- Views reports and creates new reports/user defined queries
- creates updates when necessary
- controls user passwords and usernames

Maintenance Administrator(s)

- Creates user account(s) and sets user privilege(s)
- Performs Update or Delete or Add functions
- Views reports
- Assigns staff to job(s)
- Assigns interval(s) of frequency (ies) to job(s)
- Prints reports

- Update system with job states/status
- Controls user passwords and usernames

Simple users

- View(s) reports
- Send(s) request(s)
- Sends/receives acknowledgments

Technical Staff

- Executes task(s) physically and report on job status/state

5.2 Database Specification

The database is the backend of the system that stores all data used by the system. Data from users are stored within the database. This data is then processed into a meaningful form that can be used by the user for further action.

The following are the designs of the core database tables.

Table 5.1 shows a summarized view of the kind of data required for categorization.

Classification Scheme	**Members**	**Description**
Category	Category ID Category Name Category Description	This creates the major categories for classifying assets (e.g. Vehicles, HVAC)
Type	Type ID Type Name Type Description	This further breaks downs categories into sub categories (e.g. SUV, FloorStanding)
Brand (e.g. Toyota)	Brand ID Brand Name Brand Description	This segments assets by manufacture (e.g. Toyota, LG)
ItemClass(Group)	ItemClassCode Type ID Brand ID	This groups the Category, Type and Brand together to form a Class of an asset
Location	Location Code Block ID Floor Room Number	This stores the location of each asset
ItemSpecification	Asset ID ItemClassCode Location Code	This stores information about a specific asset. It is uniquely identified by its Asset ID

Table 5.1 Summarized view of data required for categorizing assets

Table 5.2 shows a summarized view of the tables that are used in maintaining staff information and assignment. Assignment of staff is based on the skills of the staff, and the work-shift of the staff.

Classification Scheme	**Members**	**Description**
Staff	Staff ID Surname OtherNames Address	This gives a general information about staff
StaffSkill	StaffSkillID Staff ID SkillID	This maps a member of staff's unto his competencies (skills)
Skill	Skill ID Skill Name Skill Description	This contains a list of competencies that a member of staff may possess or is required for servicing an asset
StaffShift	Staff ID Shift Code	This maps staff unto their work shifts
Shift	Shift Code Begin Time End Time	Stores a list of available work-shifts

Table 5.2 Summarized view of data required for staff assignment

Table 5.3 lists the tables that are used in generating planned maintenance schedules. The Maintenance Schedule is linked to the Service by the Service ID. The Service Work-Plan is also linked to the type of service by the Service ID. The work-plan lists the steps required for servicing an asset

Classification Scheme	Members	Description
Maintenance Schedule(Planned)	Schedule ID Asset ID Service ID Start Date Frequency Next Due Date ...	This contains scheduling information for each asset
Service	Service ID Service Name Service Description	This contains a list of services required for preventative maintenance
Service WorkPlan	Service ID Service Step Number Service Task Remarks	This lists the steps (checklist) for performing a service

Table 5.3 Summarized view of data required for automatic scheduling

In Table 5.4, the history of assets that have undergone maintenance are stored in the History Table. Staff that performed those maintenance are listed in the HistoryStaff table. The History Table and the HistoryStaff table are linked via the HistoryID

Classification Scheme	Members	Description
History Table	History ID Date ItemClassCode AssetID ServiceID DateDue HoursOverDue MaintenaceCost DateOfAssignment ScheduleID RequestID	This stores the maintenance history of all asset that have been serviced and associated cost
HistoryStaff	HistoryID StaffID SkillID	Keeps the history all staff that have performed a maintenance task

Table 5.4 Summarized view of data required for tracking the history of maintenance

For users to send unplanned maintenance request, their credentials are stored in the UsersTable. This facilitates communication between the maintenance administrator and the user. The WorkStauts table keeps stores the messages sent between the user and maintenance administrator or the system. Common faults are retrieved by users for the faults table. And the request is scheduled in the unplanned maiantenace schedule. The SenderID of theWorkStatus links it to the UsersTable's UserID. The unplanned maintenance schedule is linked to the Fault table via the Fault ID. Table 5.5 shows the tables used in this interaction.

Classification Scheme	Members	Description
UsersTable	User ID User Name Password ...	This contains the information about the user or department making the request
WorkStatus	Sender ID Recipient ID Title Message Body Message Type ...	This stores the messages exchanged between the requesting party and the maintenance administrator.
UnplannedMaintenance Schedule	Request ID Date Of Request Status Asset ID Fault ID Description User ID	This contains the list of all the unplanned maintenance requests by the users or departments. The status of the request is also indicated. Type of fault is also indicated by the requesting party
Fault	Fault ID Fault Name Description	This contains a list of common faults that a user or department can select from or a custom fault that a user can specify

Table 5.5 Summarized view of data required for maintenance requests

Figure 5.1 shows the essential tables used by the software and their linkages.

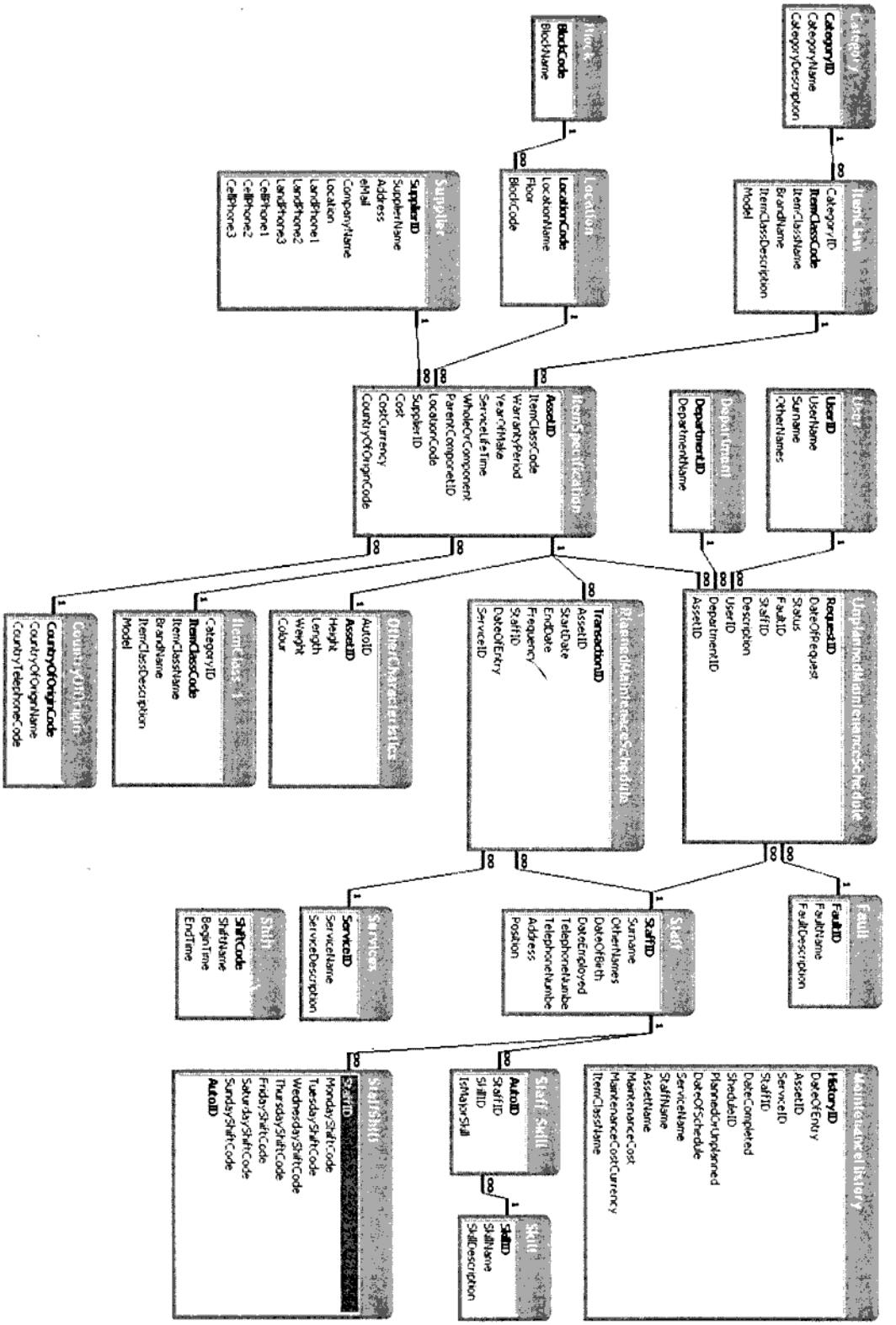


Figure 5.1 Database tables and their linkages (relationships)

6 DESIGN AND IMPLEMENTATION

6.1 Choice of Operating System Platform, Programming Language and Database Sever

Two operating system platforms were considered .The first is the Linux operating system and the second Microsoft Windows. Microsoft Windows operating system was chosen due to its wide user base and the relatively higher availability of software development tools for its environment. Linux is much more complex to administer and is not as widely used as Microsoft Windows.

6.2 Choice of Database Server

Three database servers were considered. These are Microsoft Access, Oracle and SQL Server 2005.

Microsoft Access was not chosen because it cannot handle a large number of connections and does not scale very well as the database grows.

Oracle requires quite a considerable amount of technical knowledge before use. Sybase is not very user friendly.

Microsoft SQL Server 2005 Express edition was chosen because it is stable, easy to install and administer and it is available for free and it is easier to use. It also does not require a high learning curve. It was also chosen because it is scalable and robust.

6.3 Choice of Programming Language

Microsoft Visual Basic 2005 (Express Edition) was the choice of programming language for development. It is an object oriented programming language and well suited for rapid application development. It is a powerful tool for developing Graphical User Interface applications (GUI). It is also an event driven programming language. A method can be coded to handle the events. Visual Basic 2005 (Express Edition) is relatively easy to use, stable and user-friendly. It is also available for free.

Krypton Toolkit, an extension by Component Factory written for the Microsoft Visual Basic environment, was used in many of the user interface designs. It provides a functional user interface with the support for multiple themes.

6.4 GT Maintenance Management Software

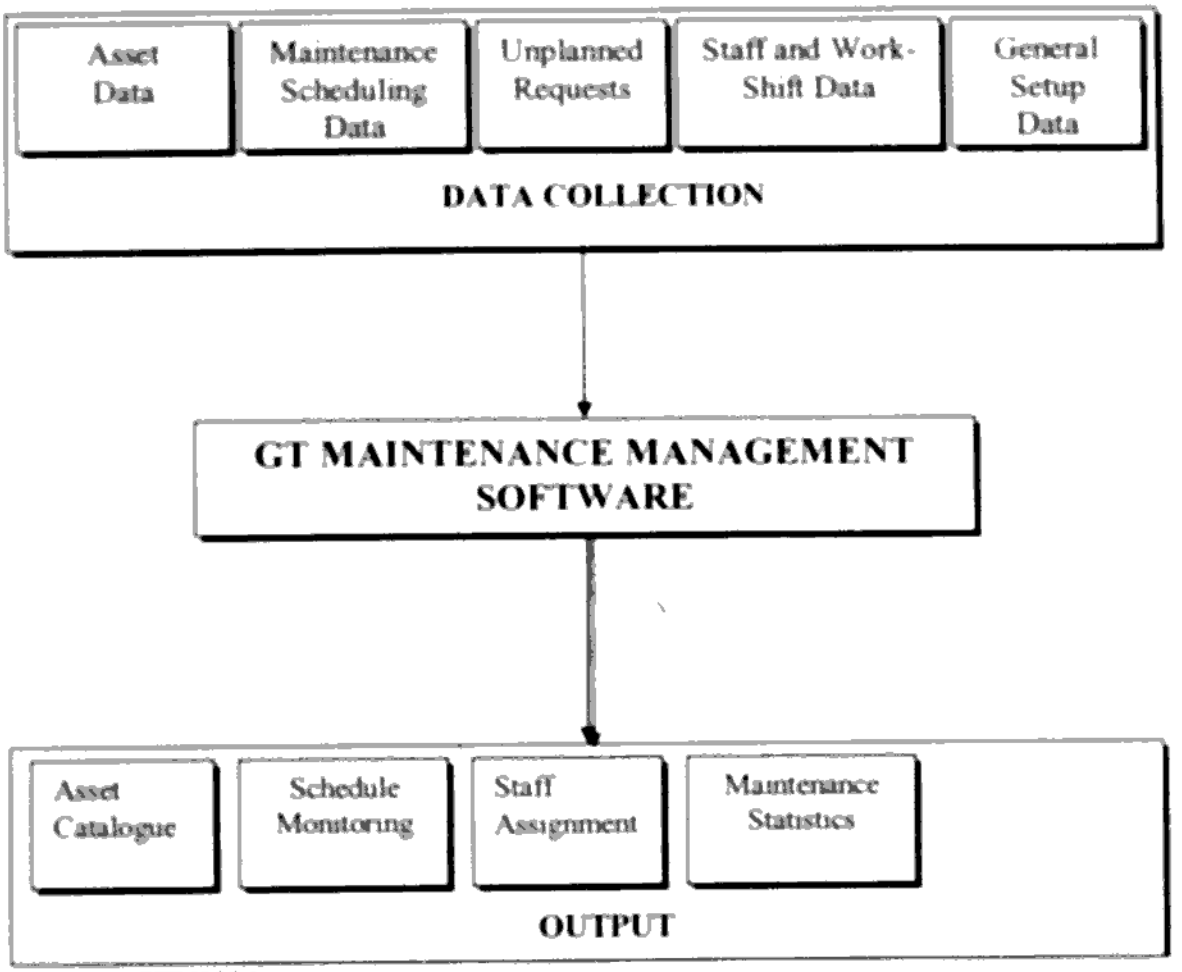


Figure 6.1 GTMMS Layout

The Layout of the GTMMS is shown in Figure 6 4 based on the software architecture discussed in Chapter 4.

The Data Collection block deals with the collection of data for cataloguing, scheduling, staff work-shifts and setting up of the software.

The GTMMS works on the collected data to produce an asset catalogue monitor maintenance schedules, suggesting and assigning staff as well as providing maintenance statistics and inventory information.

6.4.1 Organisation

The GTMMS is divided into two physically separate applications.

- The Unplanned Maintenance Client
- The Core Unit

The Core Unit is further partitioned into five major blocks or modules according to their functionalities. These are:

- Item Classification
- Item Specification
- Maintenance
- Staff
- Setup

The Unplanned Maintenance Client

This is primarily used for collecting unplanned maintenance request. The process is initiated by a user logging in with his credentials (user-name and password) as shown in Figure 6.2.

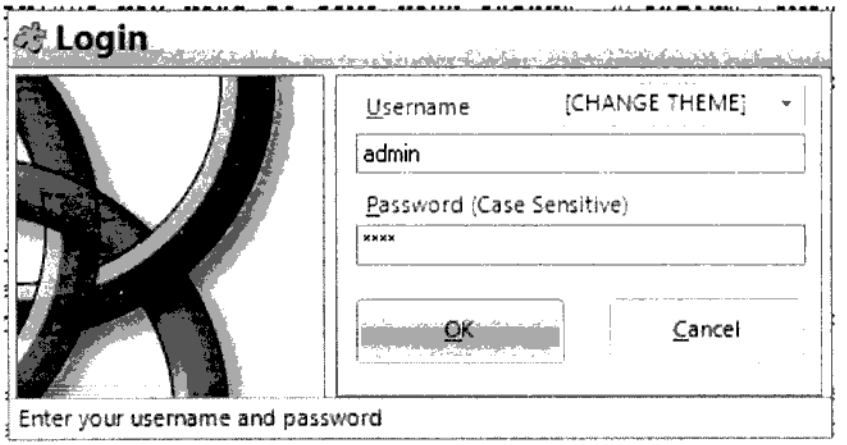


Figure 6.2 Login Screen

The user is then logged onto the Unplanned Maintenance Client. A list of previous correspondences, if any, is displayed on an email-like form as shown in Figure 6.3.

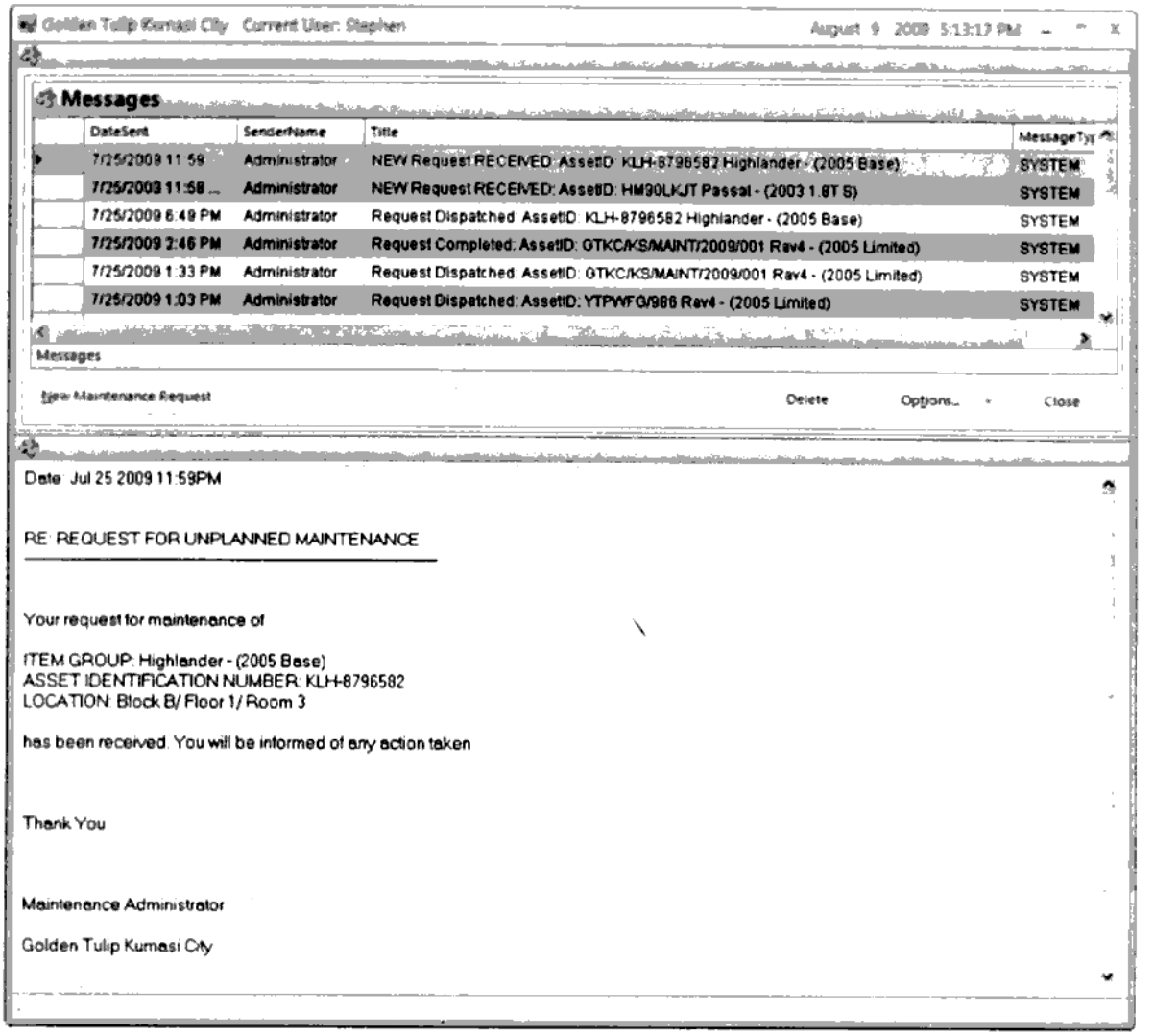


Figure 6.3 Unplanned Maintenance Client

To place a new request, the user first locates (Figure 6.4) the room and selects the item (asset) about which he wishes to report the defect. Selecting a location automatically displays all the (catalogued) assets stored in that location.

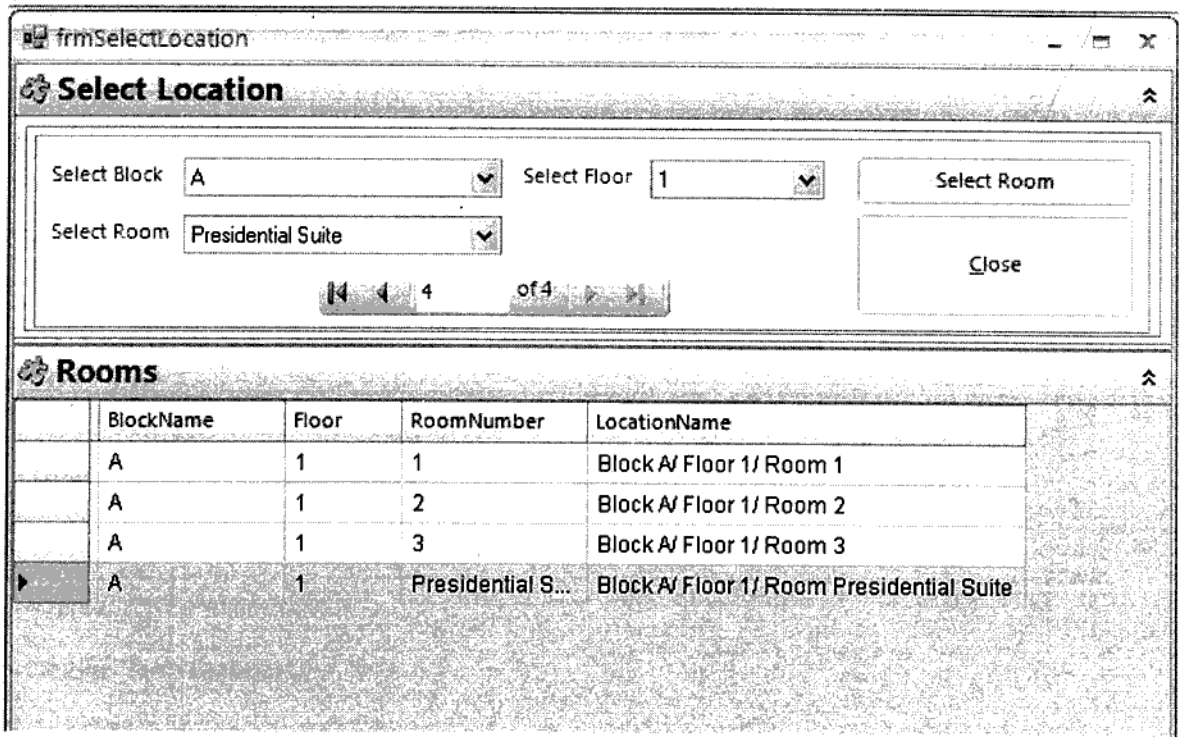


Figure 6.4 GTMMS Room Locator

The user can then select the type of fault and optionally, add a description of the problem for submission (Figure 6.5). The submitted request is then acknowledged automatically of the receipt (Figure 6.6)

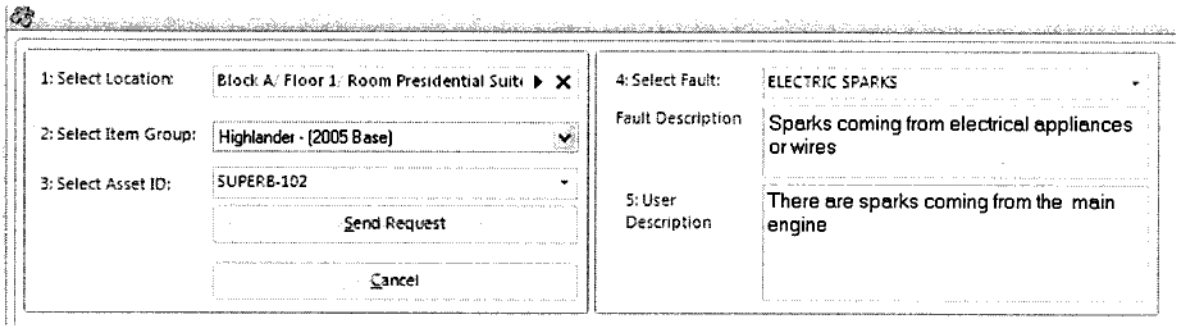
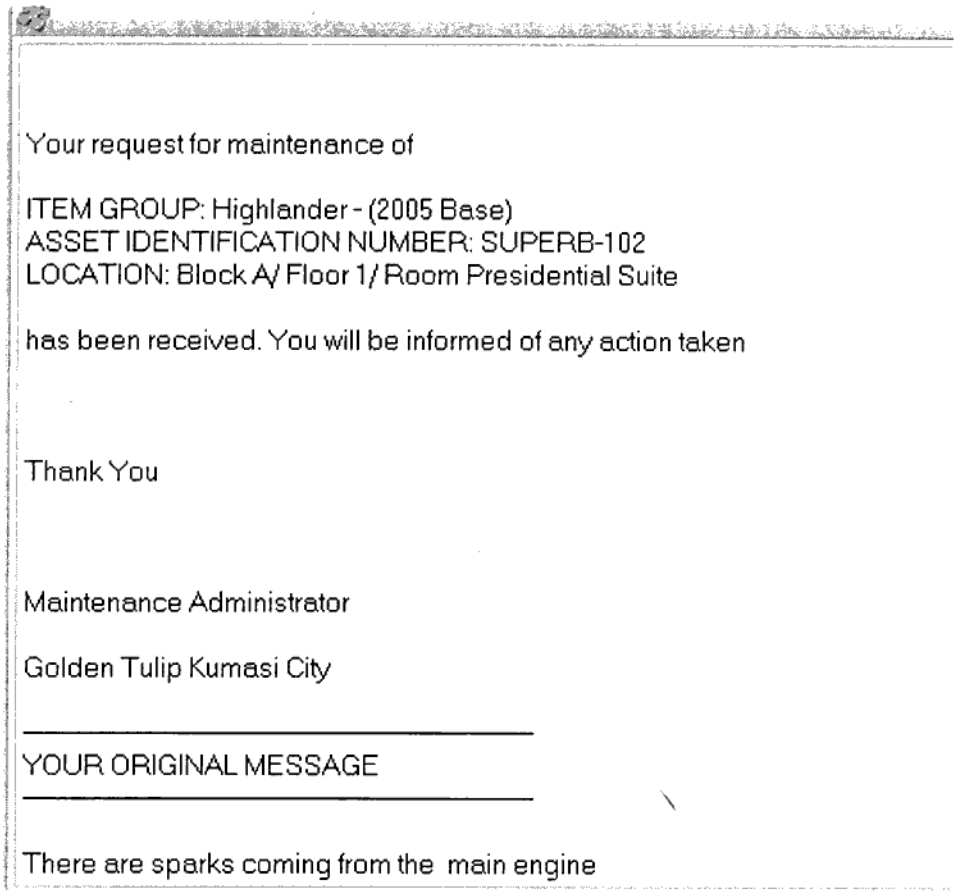


Figure 6.5 Fault submission


Your request for maintenance of

ITEM GROUP: Highlander - (2005 Base)
ASSET IDENTIFICATION NUMBER: SUPERB-102
LOCATION: Block A/ Floor 1/ Room Presidential Suite

has been received. You will be informed of any action taken

Thank You

Maintenance Administrator

Golden Tulip Kumasi City

YOUR ORIGINAL MESSAGE

There are sparks coming from the main engine

Figure 6.6 Acknowledgement of unplanned maintenance request

The rest of the unplanned maintenance processing is taken up by the Maintenance Manager.

The Core Unit

The Setup block

The Setup block provides the functionality of setting up the GTMMS for use. It comprises the Room Number Generator for generating room numbers, Services and Faults setup for predefining a list of services for scheduling maintenance activities, Service Work-Plan for setting up the checklist for a maintenance task, and a User

Manager for managing the user login and access rights to the system. It also has a Supplier, Country of origin for storing supplier and country information.

Room Number Generator: The Room Number Generator works by allocating room numbers on a floor by floor basis. The room numbers may be sequential over a range of numbers, or can be named as shown in Figure 6.7.

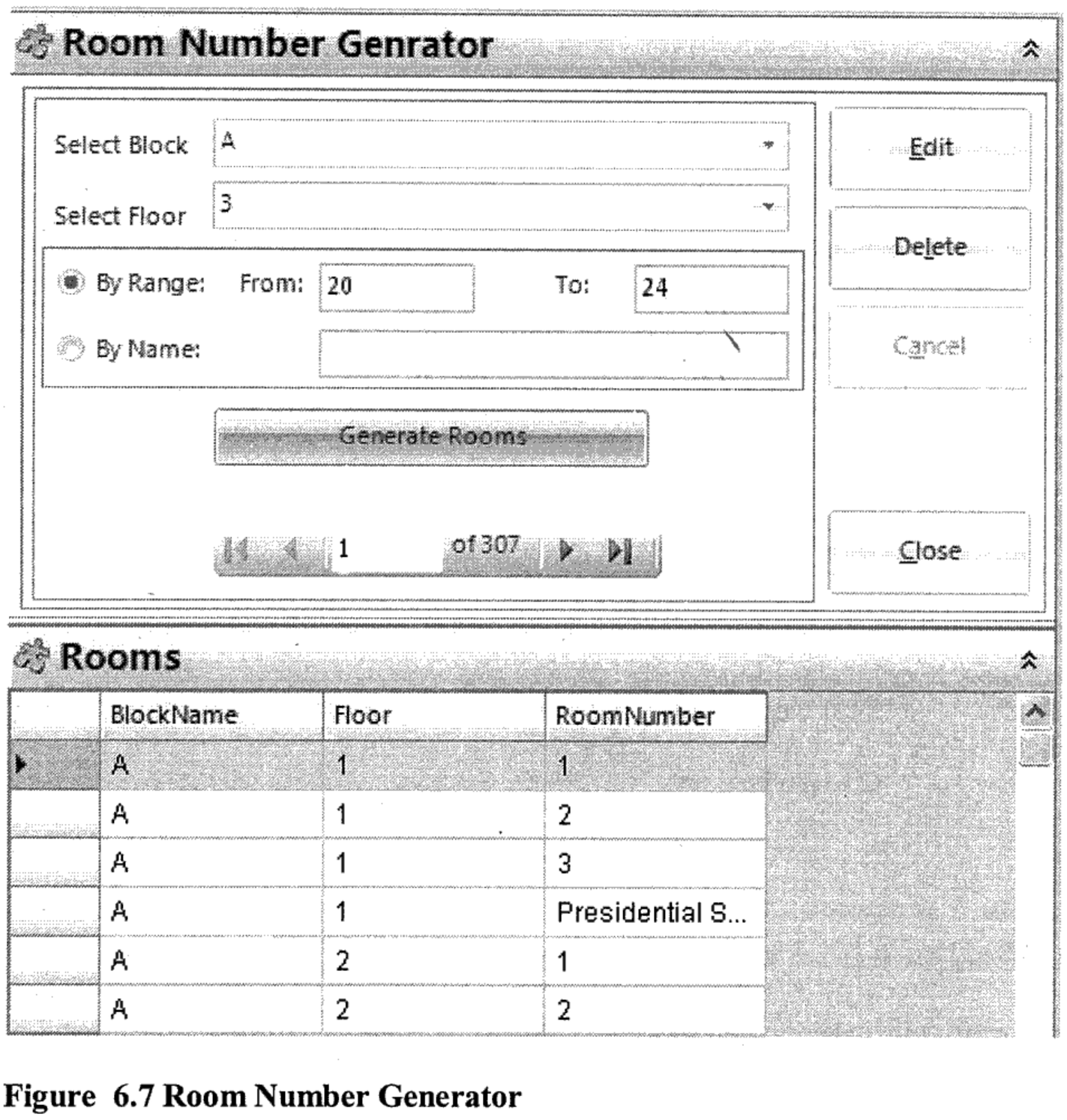


Figure 6.7 Room Number Generator

Services and Faults: These two modules are technically the same but functionally different. The Services module is for managing all services that are

Manager for managing the user login and access rights to the system. It also has a Supplier, Country of origin for storing supplier and country information.

Room Number Generator: The Room Number Generator works by allocating room numbers on a floor by floor basis. The room numbers may be sequential over a range of numbers, or can be named as shown in Figure 6.7.

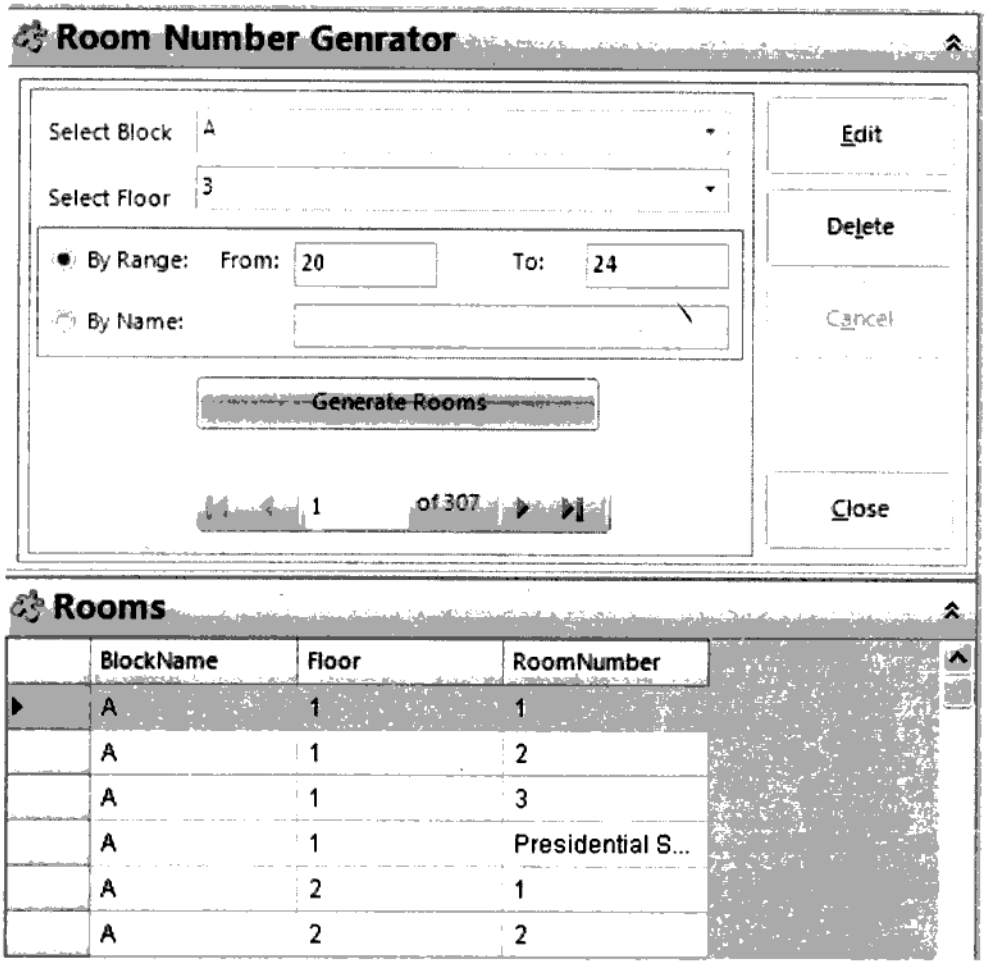


Figure 6.7 Room Number Generator

Services and Faults: These two modules are technically the same but functionally different. The Services module is for managing all services that are

necessary for planned maintenance scheduling, The Faults module is used for managing common faults that may arise from unplanned maintenance requests *(See Figure 6.8)*.

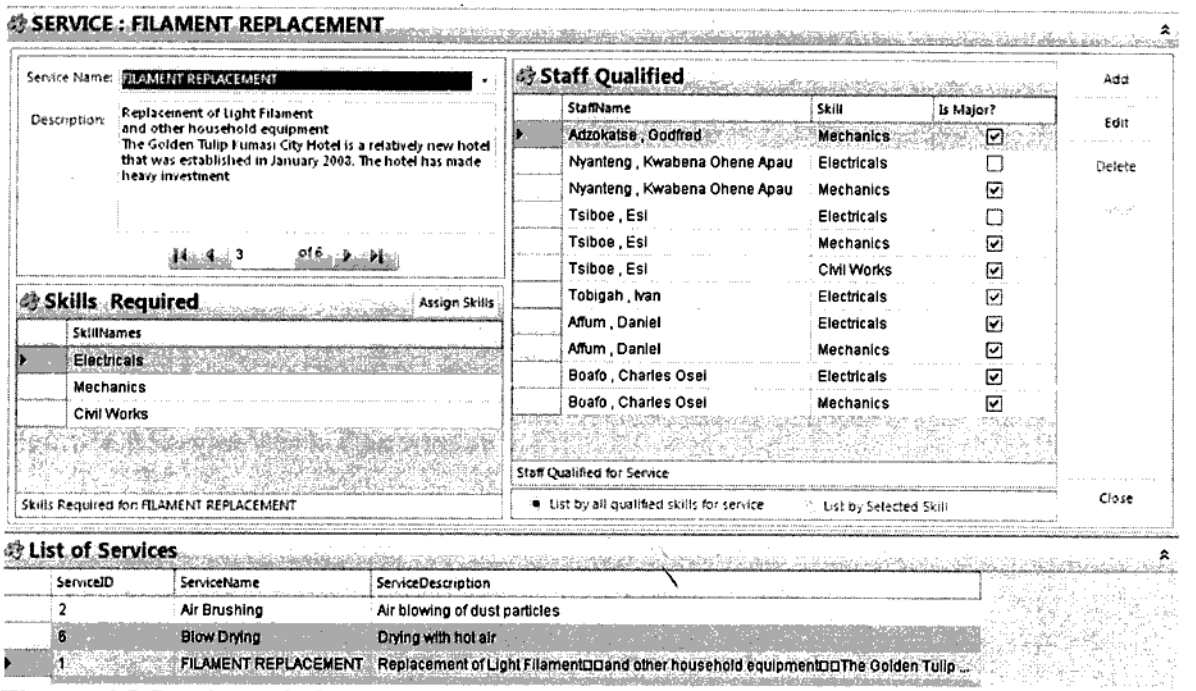


Figure 6.8 Services window

Service Work-Plan: This module manages the list of the various steps that are necessary in carrying about a service *(See Figure 6.9)*

User Manager: The User Manager (See Figure 6.10) manages the addition, modification or removal of user accounts and access rights into the GTMMS. All the passwords are stored in the database using a hash function for security

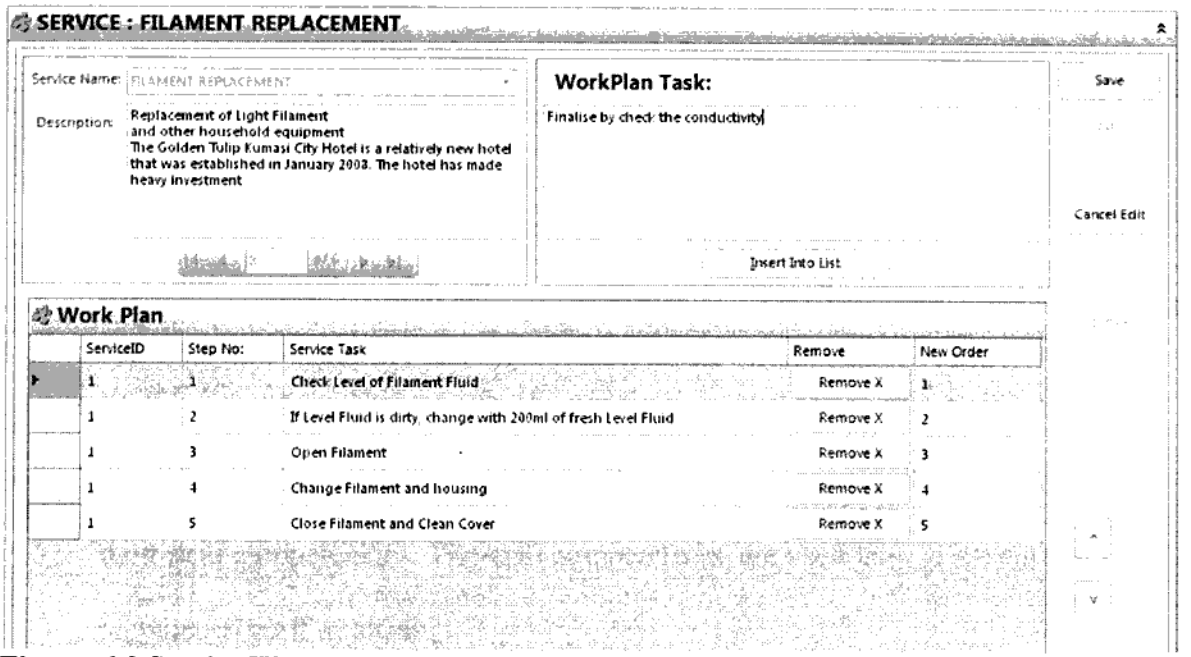


Figure 6.9 Service Work-Plan window

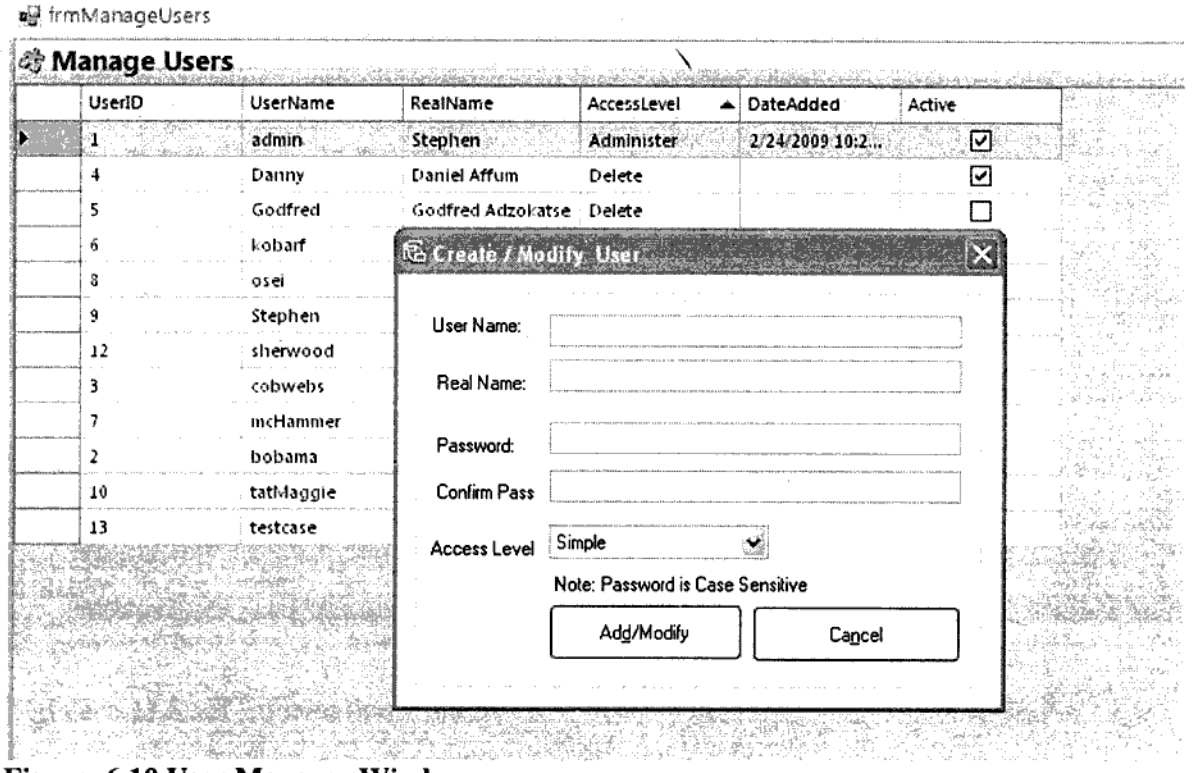


Figure 6.10 User Manager Window

Staff Block

The Staff block provides the functionality of managing the technical staff responsible for carrying out the physical maintenance tasks. It comprises the Staff information module, the Skills module, and the Work Shift module.

Staff Information: This module (Figure 6.11) is used for storing staff details such as names, date of birth and contact addresses etc. It is also used for assigning their skill competencies. The skill competencies are used later when assigning a member of staff to a task.

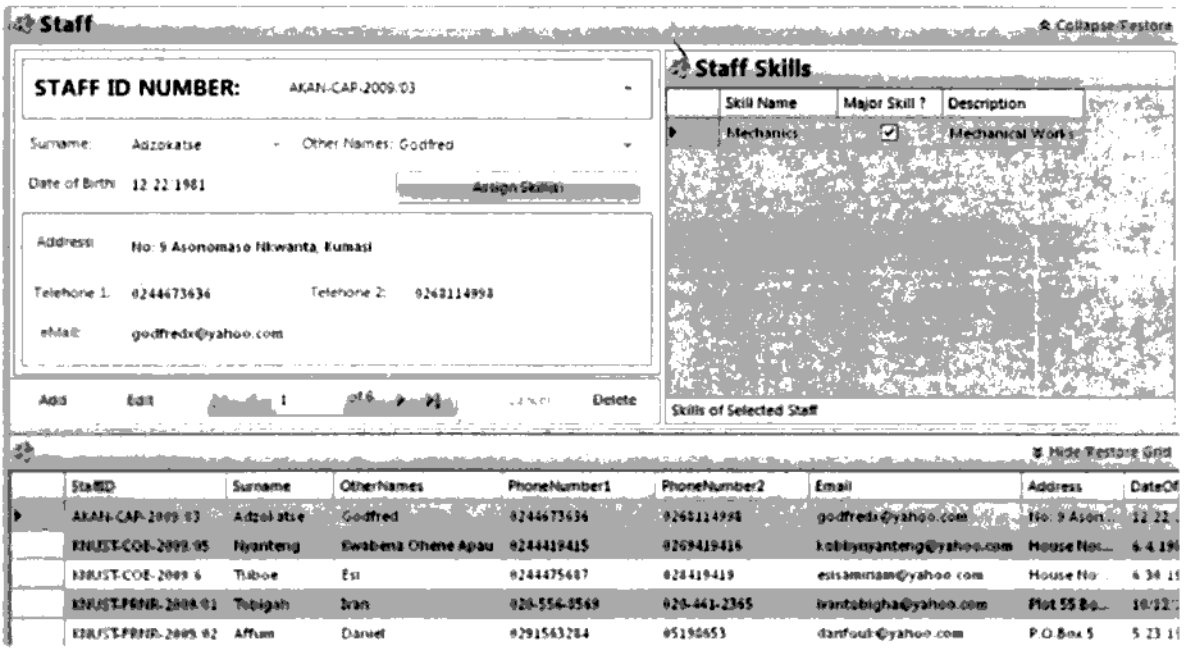


Figure 6.11 Staff Information window

Skills: The Skills (Figure 6.13) module manages the list of skills required to perform a maintenance task. The skills are then assigned to staff with that qualification.

Work Shift: This module (Figure 6.12) manages the working hours of staff. It provides the maintenance administrator the convenience being able to tell when a member of staff is available for work when scheduling.

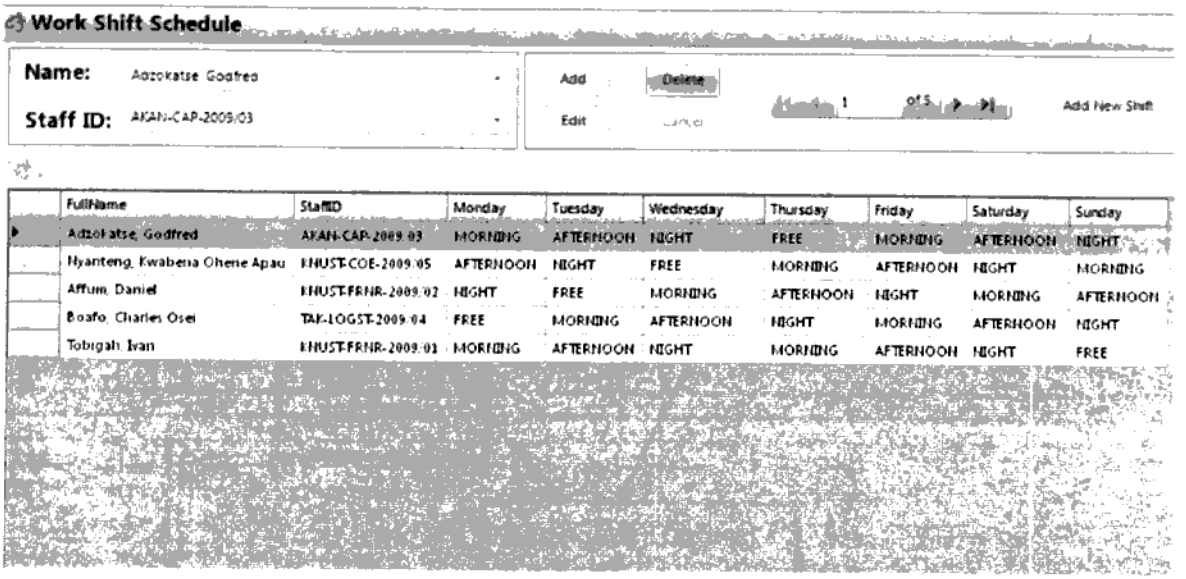


Figure 6.12 Work Shift Schedule

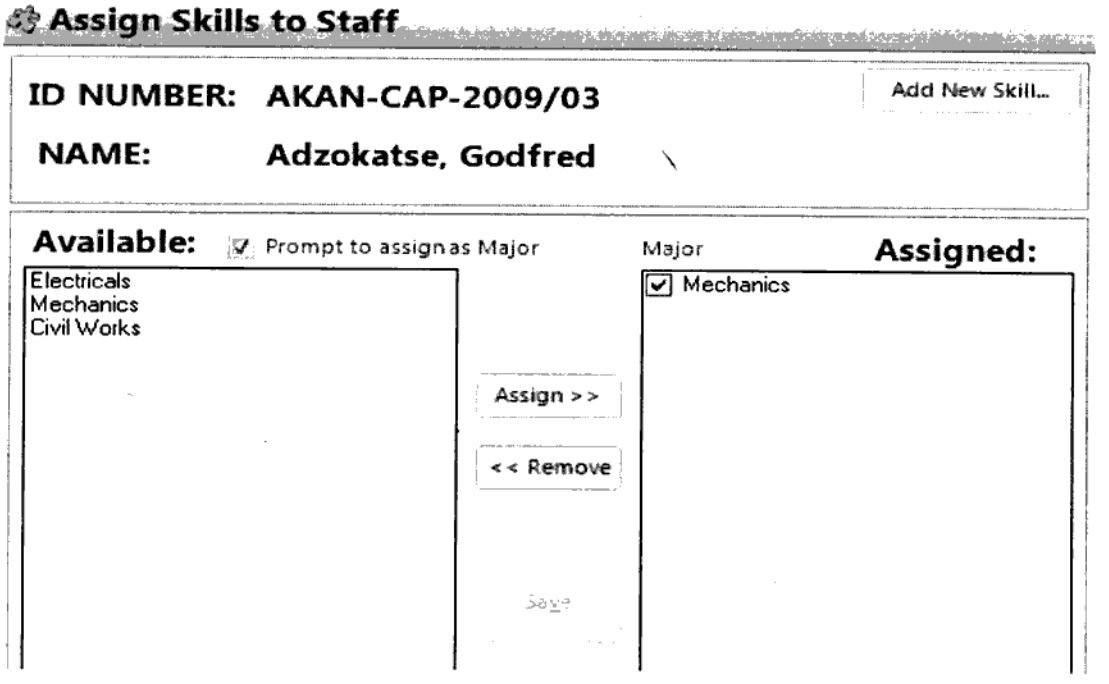


Figure 6.13 Skill Assignment window

Item Classification Block

The Item Classification block is used for setting up a categorization scheme of assets. This block does not create an inventory of assets and is solely concerned with classification. It is divided into Category, Type, Brand and Class (Group) categories (Figure 6.14).

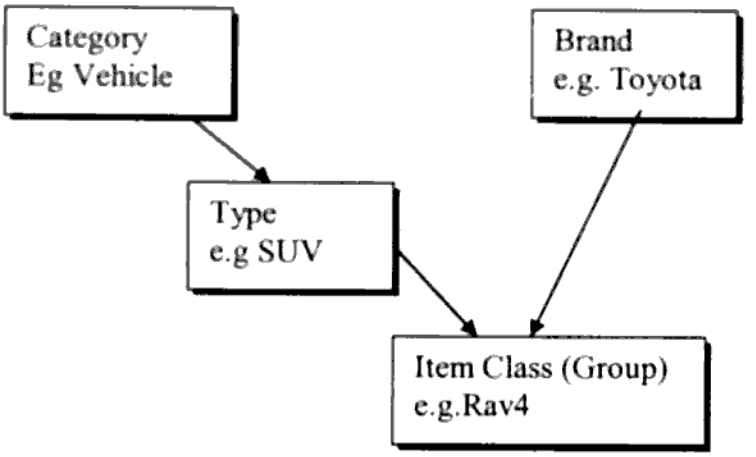


Figure 6.14 Asset Classification Scheme

Category: The Category module (Figure 6.15) is a larger grouping of assets. Assets belonging to different categories have very little in common if any at all. For example, the category "Vehicle" has little in common with the category "Furniture".

Type: The Type module (Figure 6.16) narrows down the category to those with some major characteristic in common. Items classified at the type level may have identical functionalities but different implementation or appearance. For

example, an Sports Utility Vehicle (SUV) and a Saloon car belong to the same Category (Vehicles) with identical functionalities but implemented differently.

Brand: This module groups 'Types' according to manufacturers. For example, Toyota, Ford, VW may produce items belonging to the same 'Category' and 'Type'. The Brand is independent of the 'Category' or 'Type' since one manufacturer may be producing several items in many Categories.

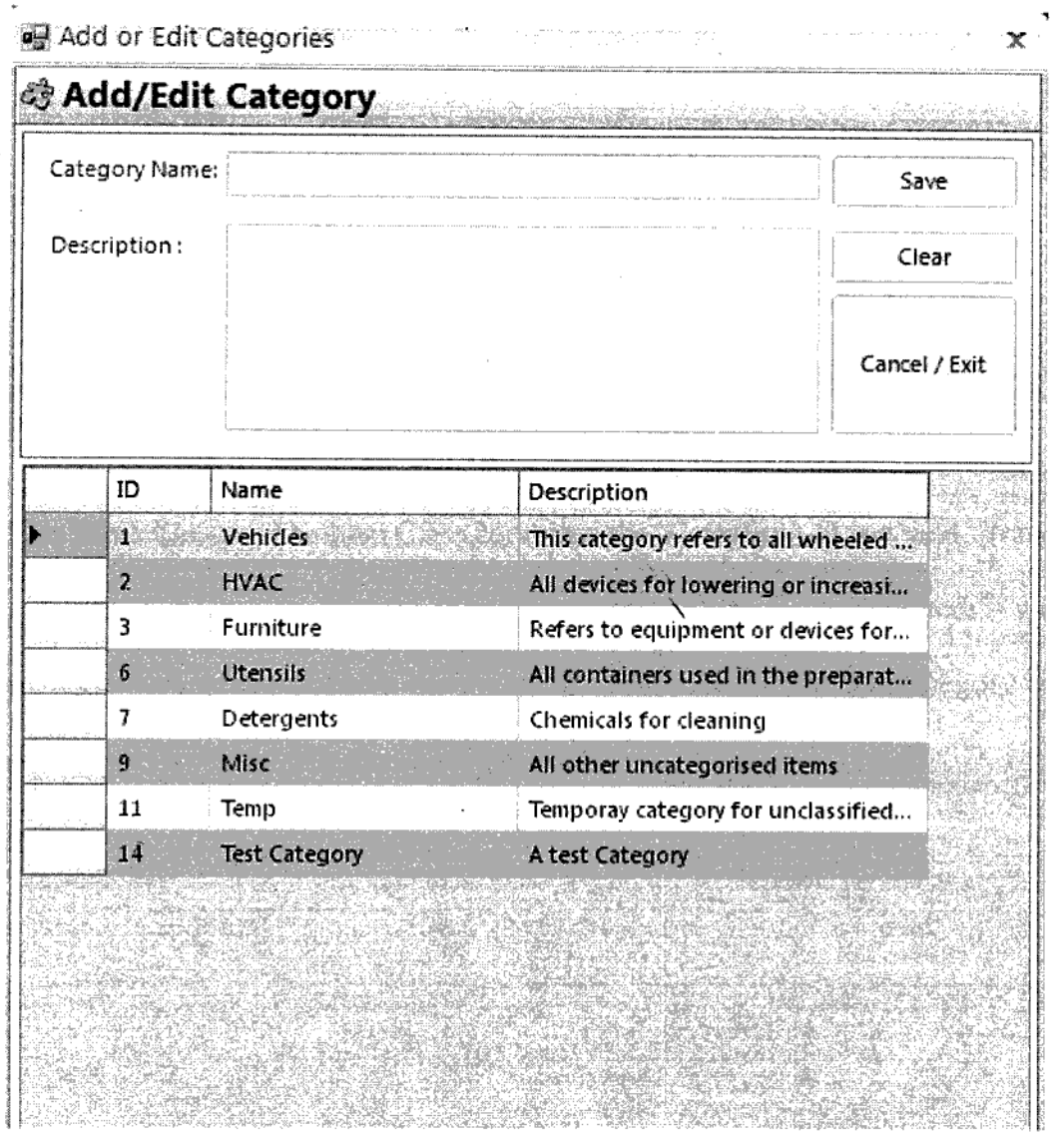


Figure 6.15 Item Category window

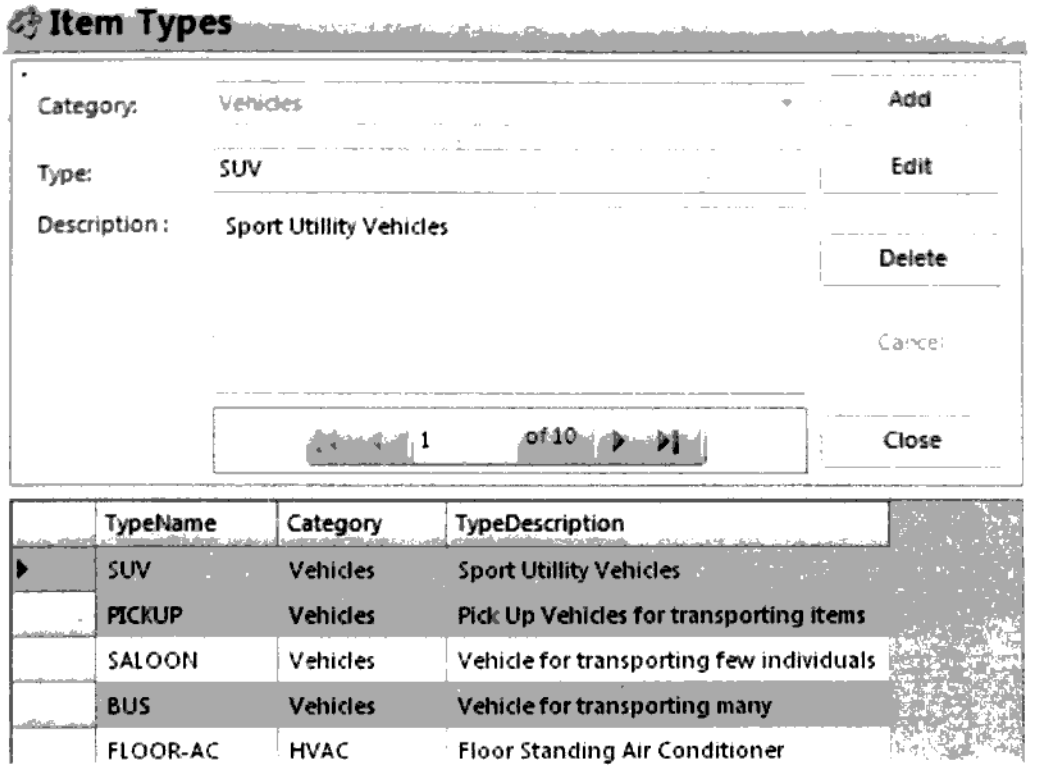


Figure 6.16 Item Type window

Item Class/Group: The Item Class combines the 'Category', 'Type' and 'Brand' into a group item. It also classifies the group item into Models. This group item is what individual assets derive from. For example, a group item Rav4 is of category vehicle, of type SUV and brand Toyota *(See Figure 6.17)*

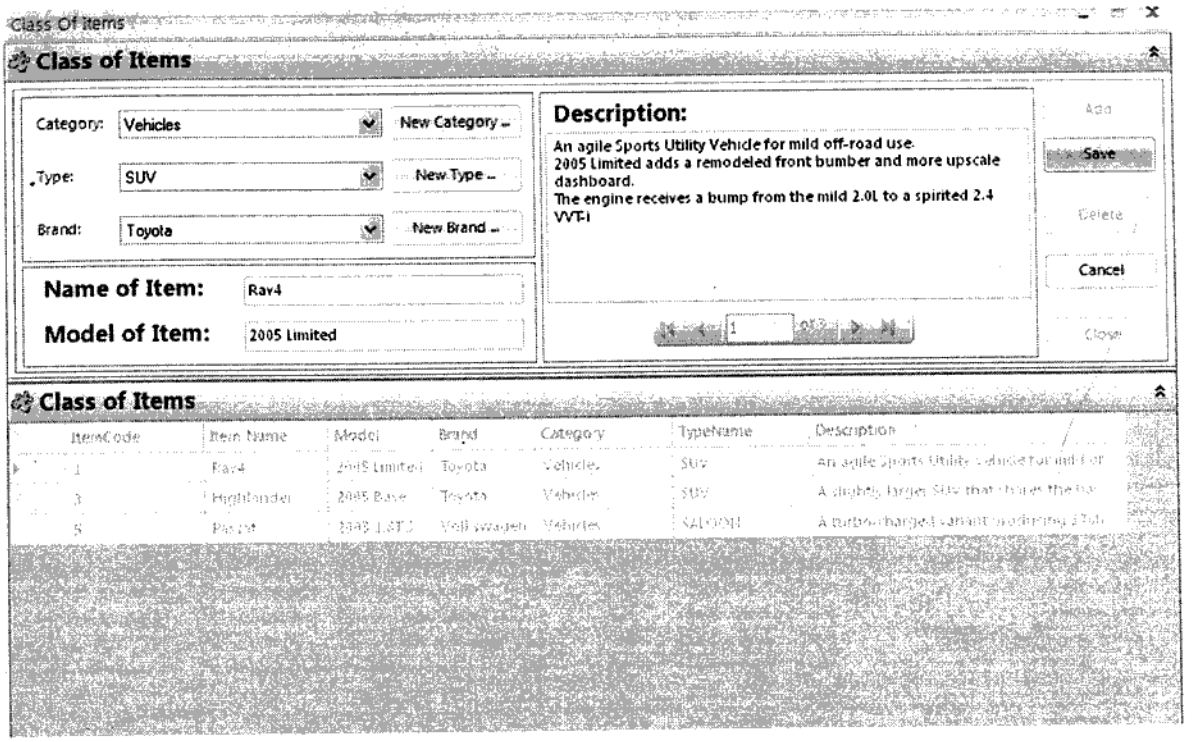


Figure 6.17 Item Class window

Item Specification block

The Item Specification module (Figure 6.18) is the only component of this block. It is used for cataloguing of assets. Details of the asset are entered into the system by choosing the correct item class and assigning its asset identification number and/or serial number. The Item Specification module is an inventory of asset. Some details include the warranty expiration date, location, supplier and remarks about the age of the item.

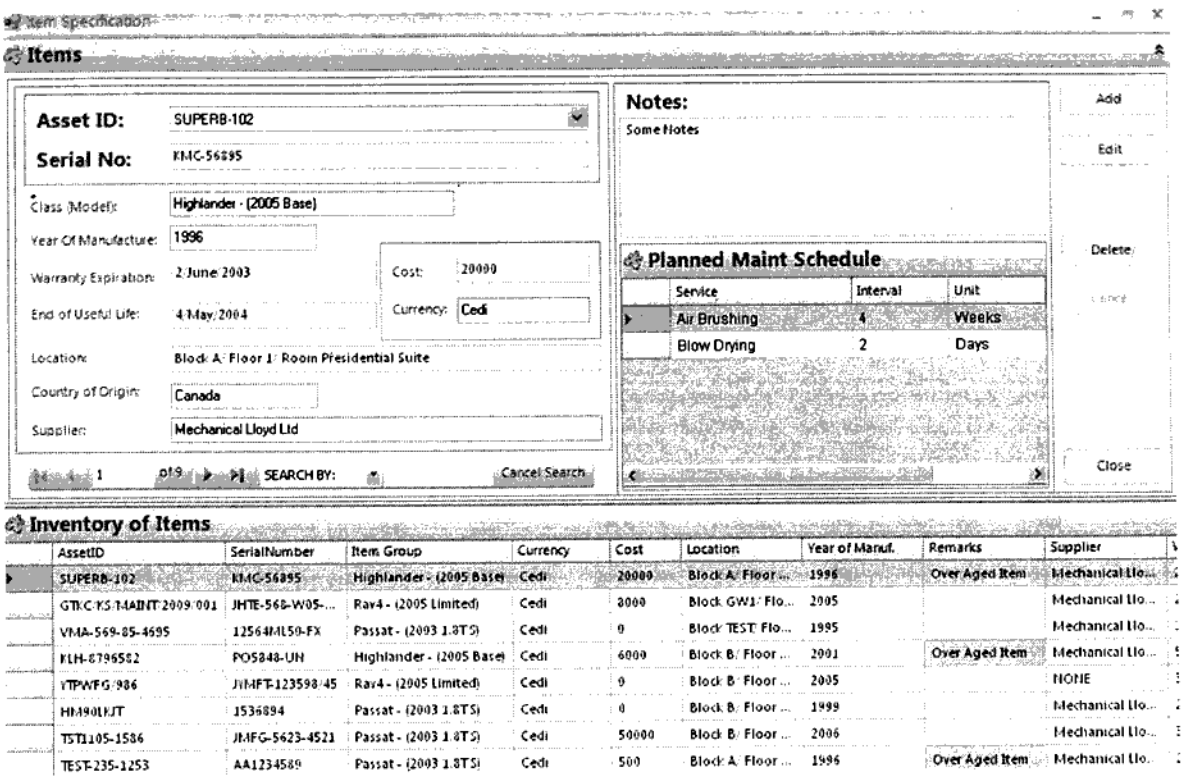


Figure 6.18 Item Specification window

Maintenance block

The Maintenance block brings together all activities related to maintenance. It consists of an Individual Scheduler module, a Group Scheduler module, a Planned Maintenance Manager module, Unplanned Maintenance module and Maintenance reports modules. The Maintenance block constitutes the Maintenance Manager.

Individual and Group Schedulers: These are responsible for assigning planned maintenance schedules of various services to assets either individually or as groups *(See Figures 6.19, 6.20).*

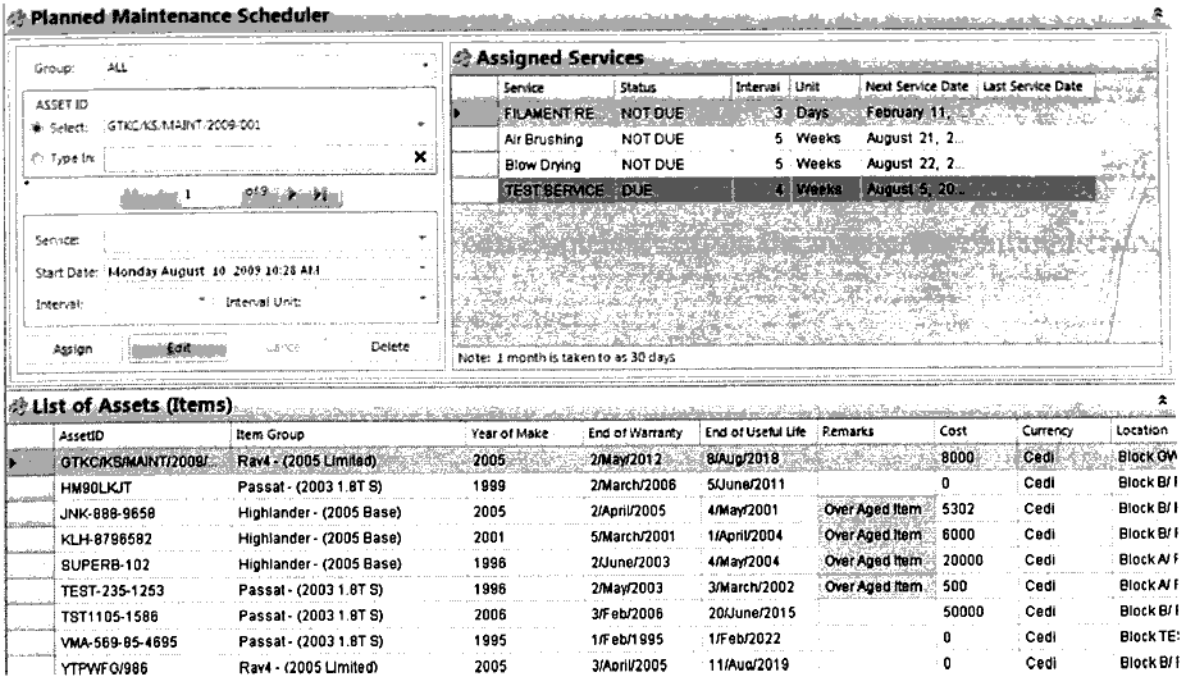


Figure 6.19 Individual Scheduler

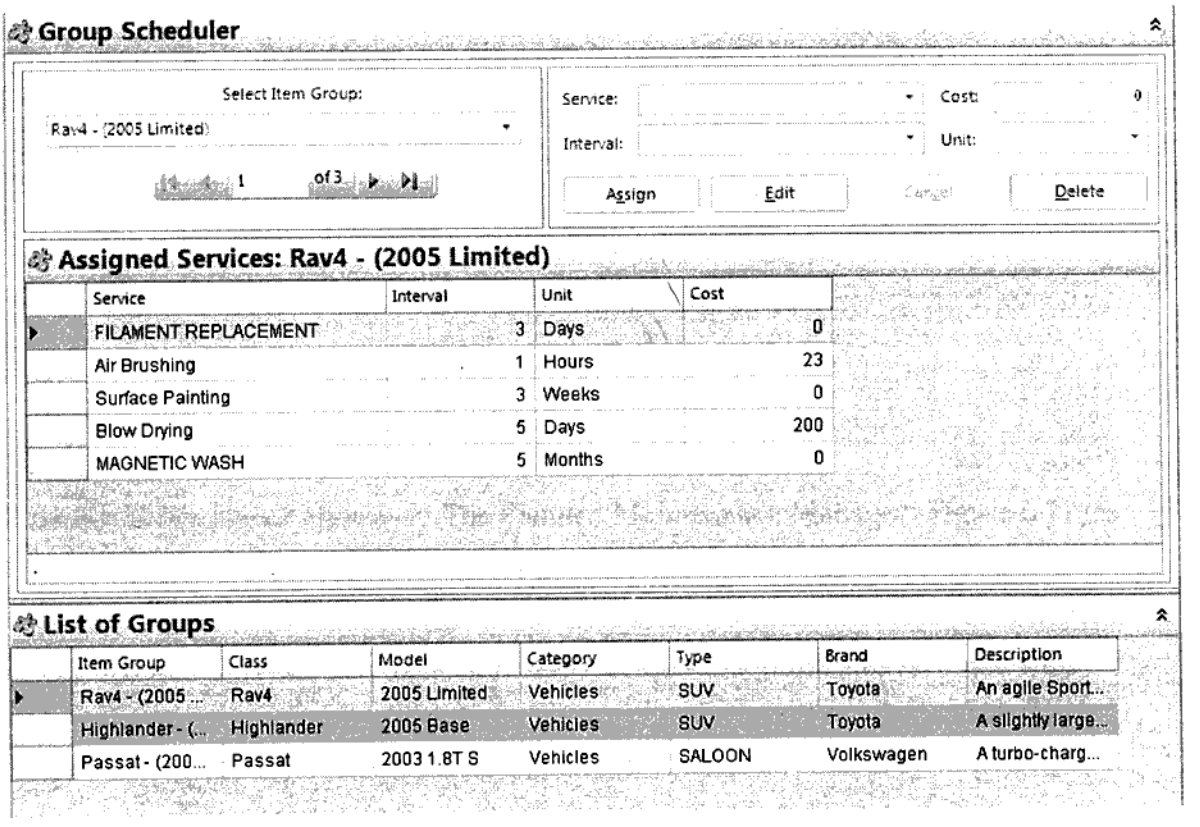


Figure 6.20 Group Scheduler

Unplanned Maintenance Manager: This module (Figure 6.21) manages the requests that fall outside planned maintenance activities. The user requests coming from the Unplanned Maintenance Client is received by this manager. Staff are then assigned to the task and dispatched. All correspondences from the maintenance administrator to the user is also done via this manager.

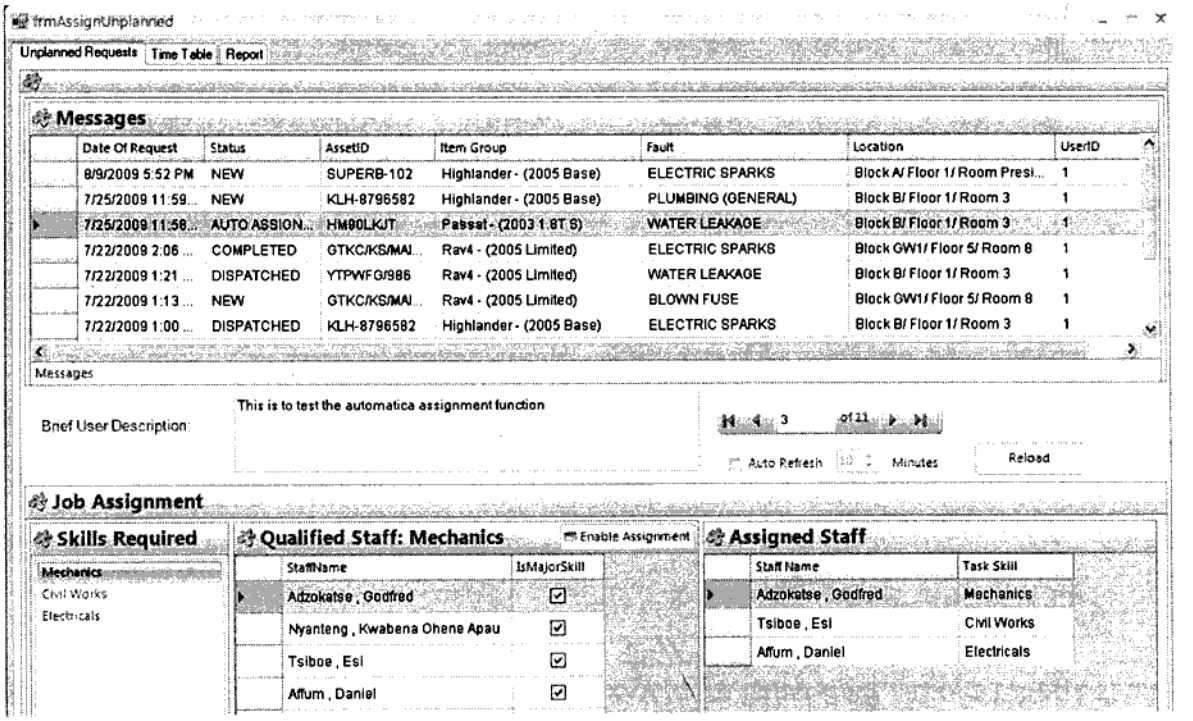


Figure 6.21 Unplanned Maintenance Manager

Planned Maintenance Manager: The Planned Maintenance Manager (Figure 6.22) is similar to the Unplanned Maintenance Manager with the exception that it takes its input from a schedule instead of a request. It also manages assignment of staff and staff dispatches.

It has a job assignment tab for assigning tasks to staff, a planned schedule tab for listing the planned schedules of assets, a list of all dispatched jobs, a service work-plan tab and a time table tab for displaying the time tables of staff.

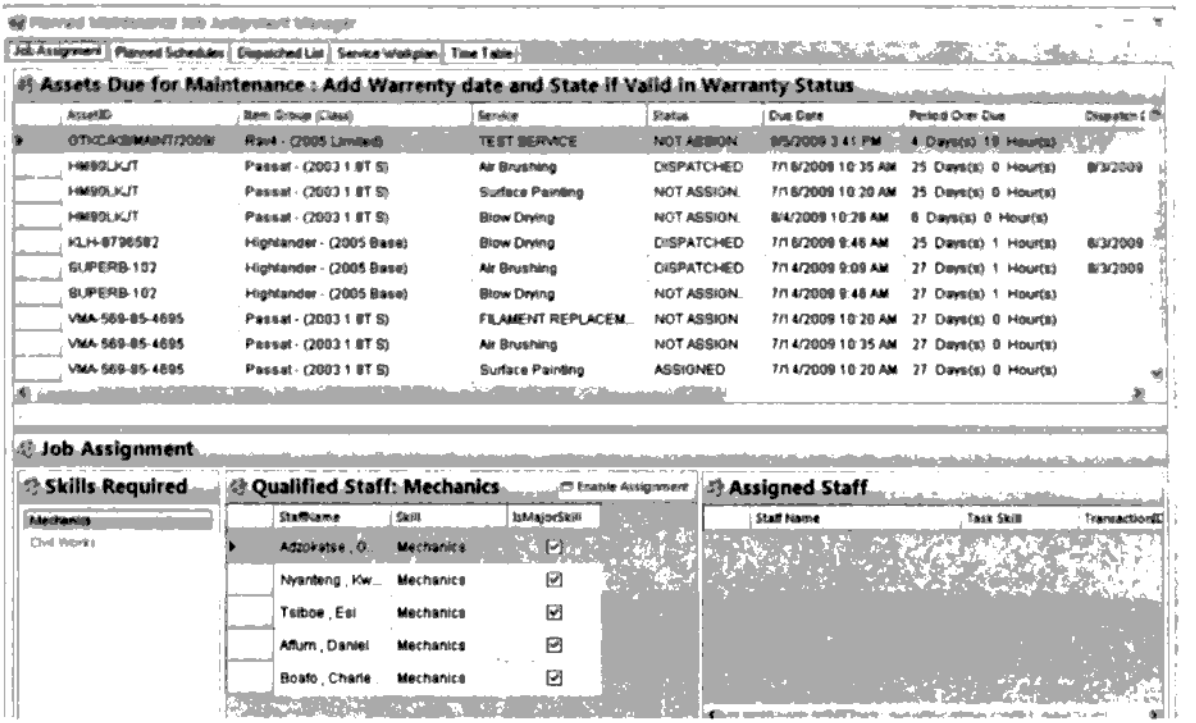


Figure 6.22 Planned Maintenance Manager

Reports: The Maintenance block contains a module for generating various reports about maintenance activities. These reports includes a maintenance summary that summarises all maintenance activities and the costs associated with such activities, a maintenance history of assets, history of staff assignment and a dispatch list etc. Some of these reports can be queried to using the criteria of interest. The reports can be exported to PDF or Excel Formats

Figure 6.23 Sample Report window

6.5 Sample Case Studies

6.5.1 Unplanned Maintenance Requests

A cleaner in one of the guest rooms notices that there are some electrical sparks coming from a television set in the room. The cleaner puts off the appliance and decided to report the incident using the Unplanned Maintenance Client.

Logging into the system brings the Unplanned Maintenance Client Message window. Notice that there are no messages in the Message window (Figure 6.24).

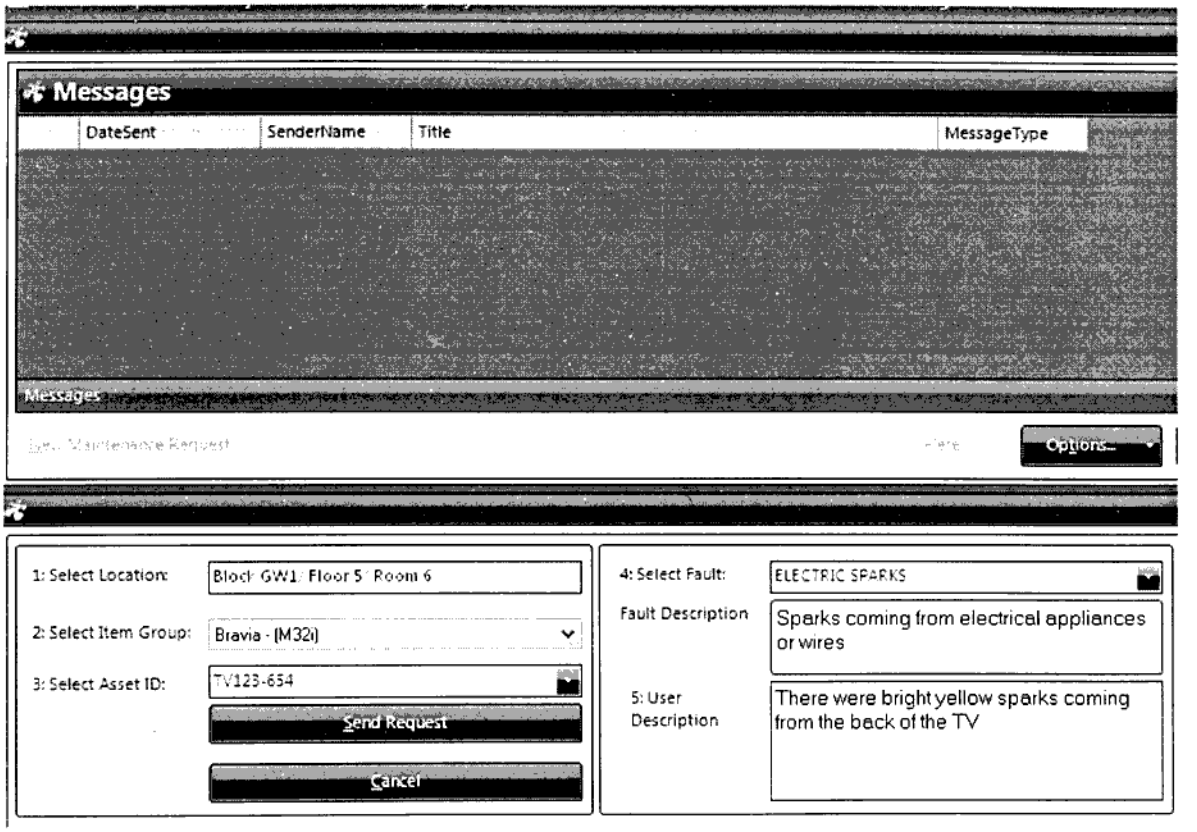


Figure 6.24 The Unplanned Maintenance Message window

On submission of the fault, a message is automatically sent to the cleaner of the receipt of message as shown in Figure 6.25.

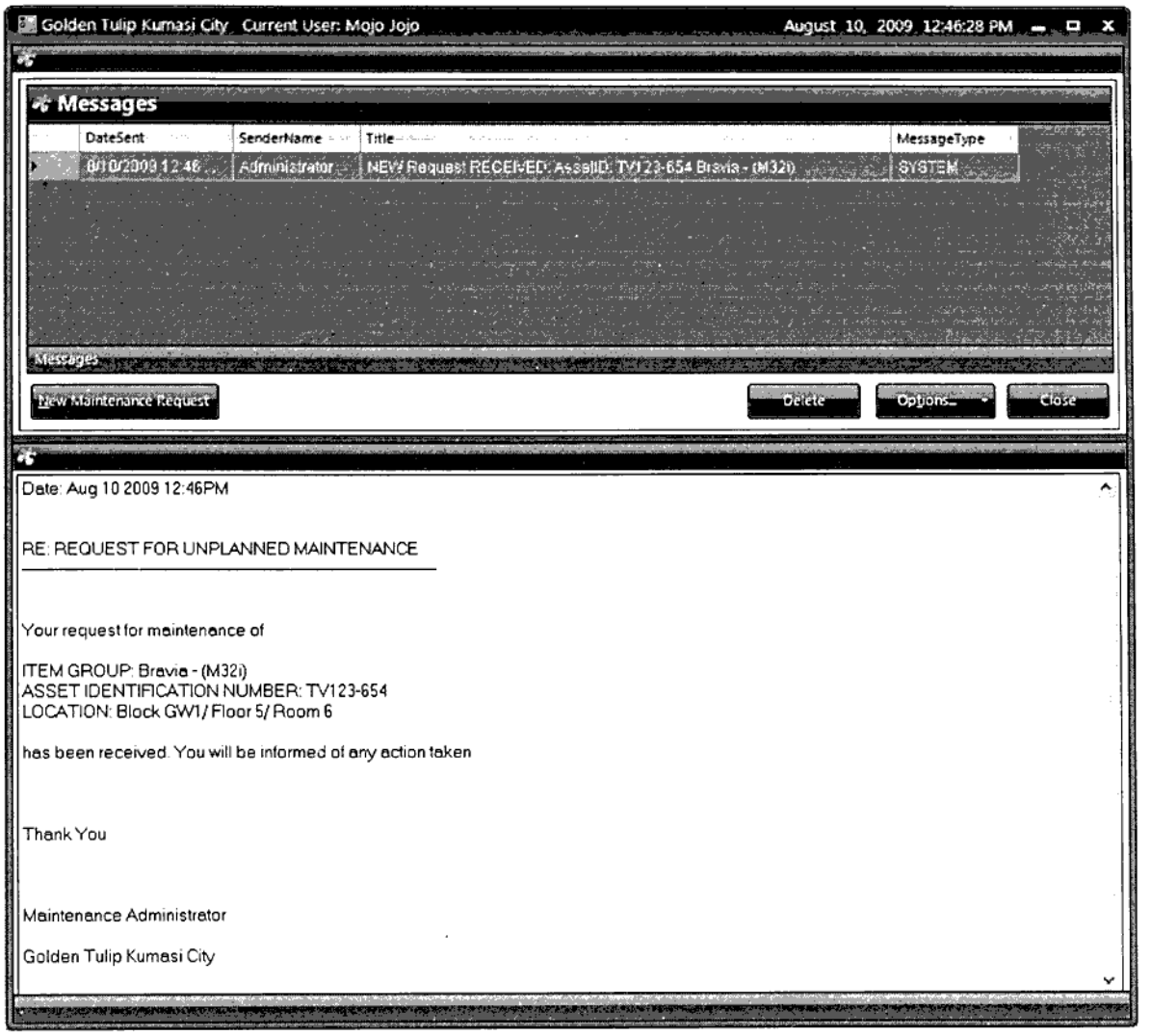


Figure 6.25 Message window displaying acknowledgement

The maintenance administrator, upon receipt of the request (Figure 6.26), reviews a list of suggestions by the system on competent staff for the job. He assigns the staff and dispatches them to the task (Figure 6.27, 6.29).

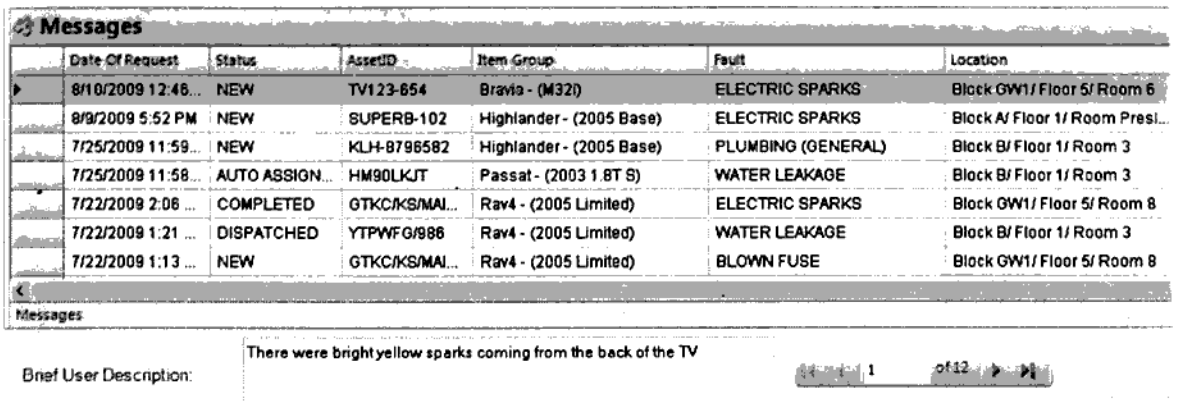


Figure 6.26 Unplanned Maintenance Manager's view of the new request

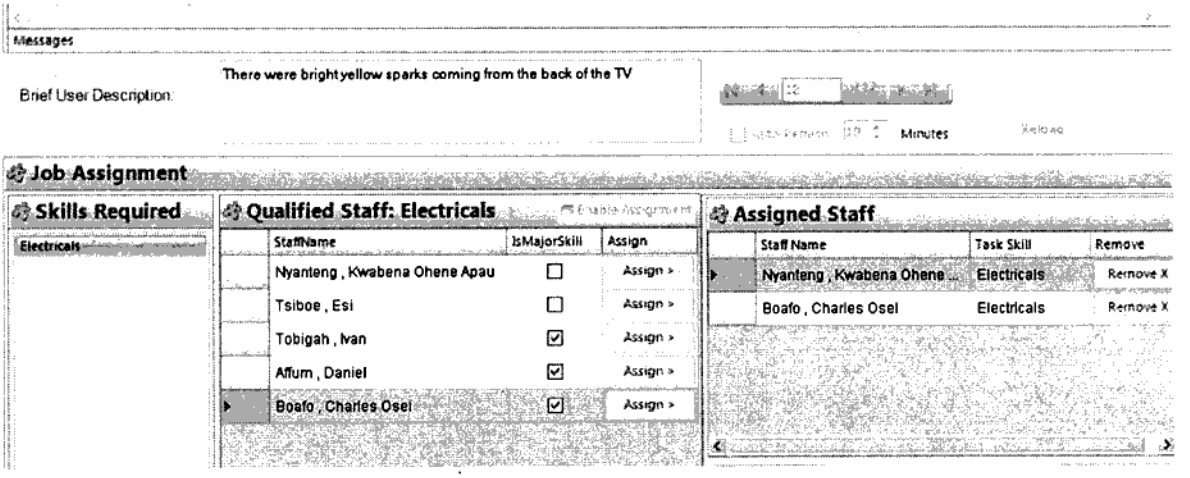


Figure 6.27 A view of competent (left) and staff assigned (right)

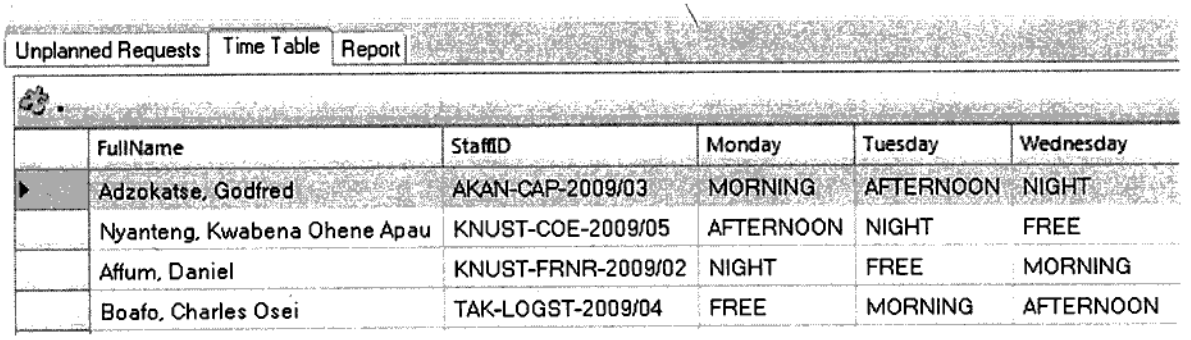
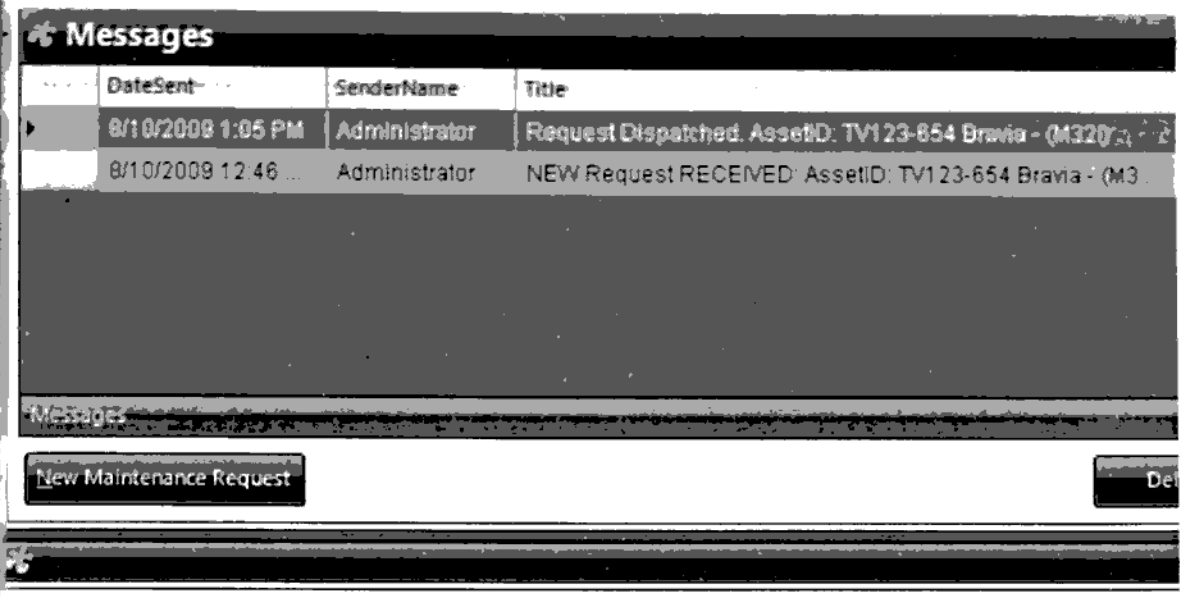


Figure 6.28 A view of the work-shift of staff



Date: Aug 10 2009 1:05PM

Staff haave been dispatched for the maintenance of

ITEM GROUP: Bravia - (M32i)
ASSET IDENTIFICATION NUMBER: TV123-654
LOCATION: Block GW1/ Floor 5/ Room 6

. You will be informed upon completetion

Thank You

Maintenance Administrator

Figure 6.29 Cleaner's notification of dispatch

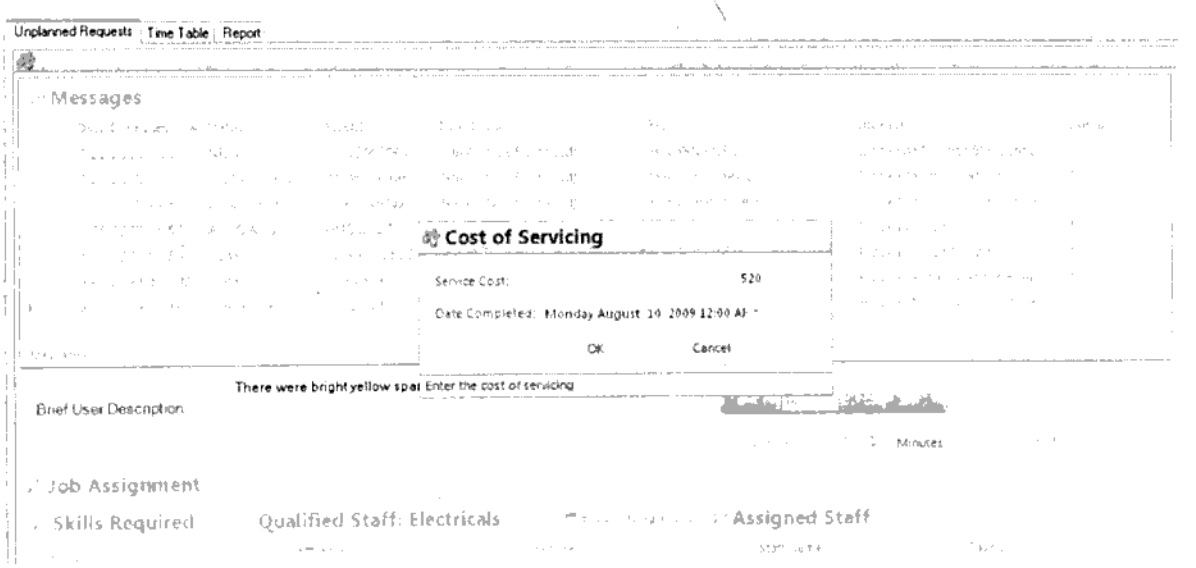


Figure 6.30 Finalizing the job when completed

Upon completion of the task, the maintenance administrator marks the job as completed with the cost involved and the date of completion as shown in Figure 6.30. The cleaner is then notified of the completion of the task (Figure 6.31).

Figure 6.31 Notification of task completion

6.5.2 Planned Maintenance Schedule

The maintenance administrator wants to schedule an item for regular preventative maintenance that will recur hourly beginning from the time of schedule. He logs into the system and starts up the Individual Planned Maintenance Scheduler (Figure 6.32).

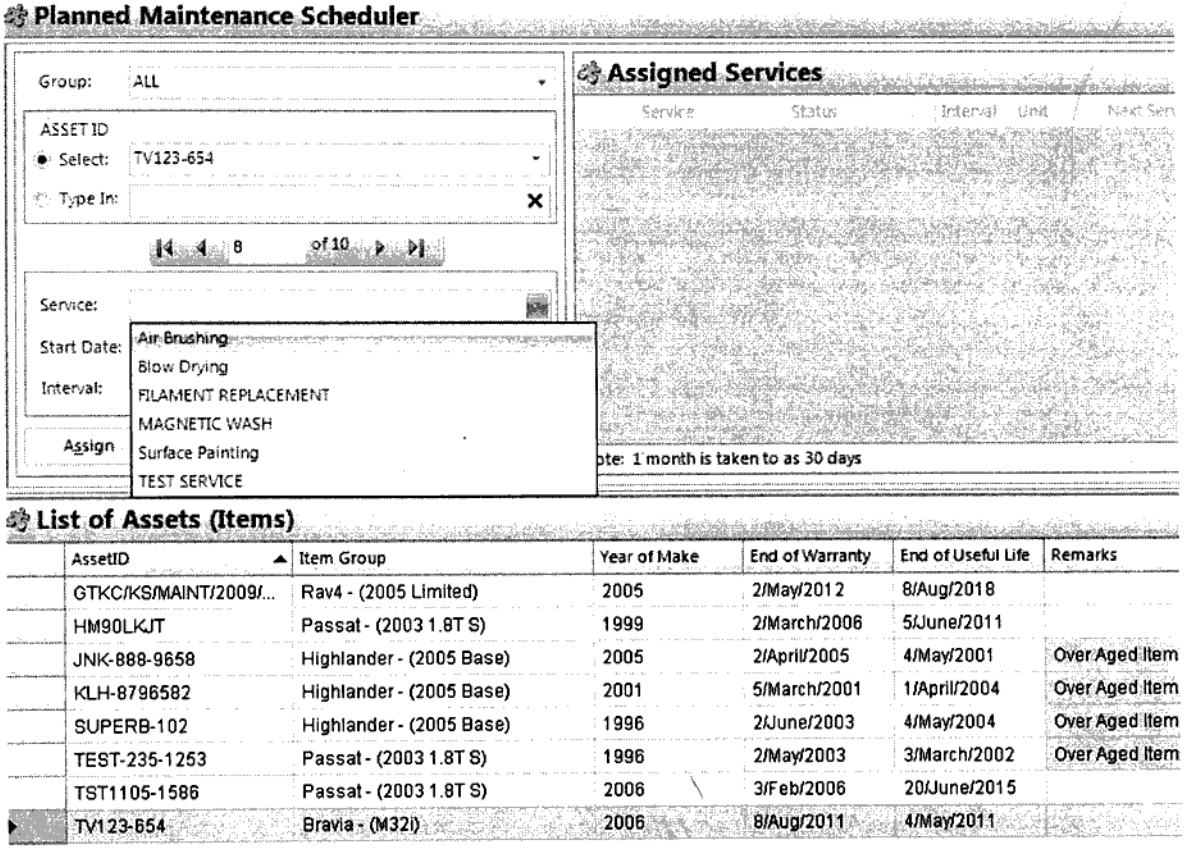


Figure 6.32 Selection of Service for scheduling

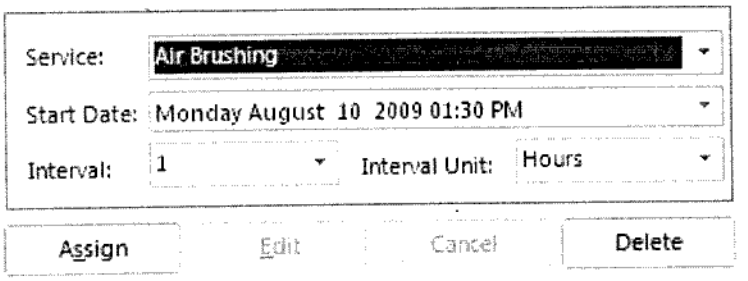


Figure 6.33 Selection of time interval for servicing

A service interval is selected by the administrator and assigned (Figure 6.33)

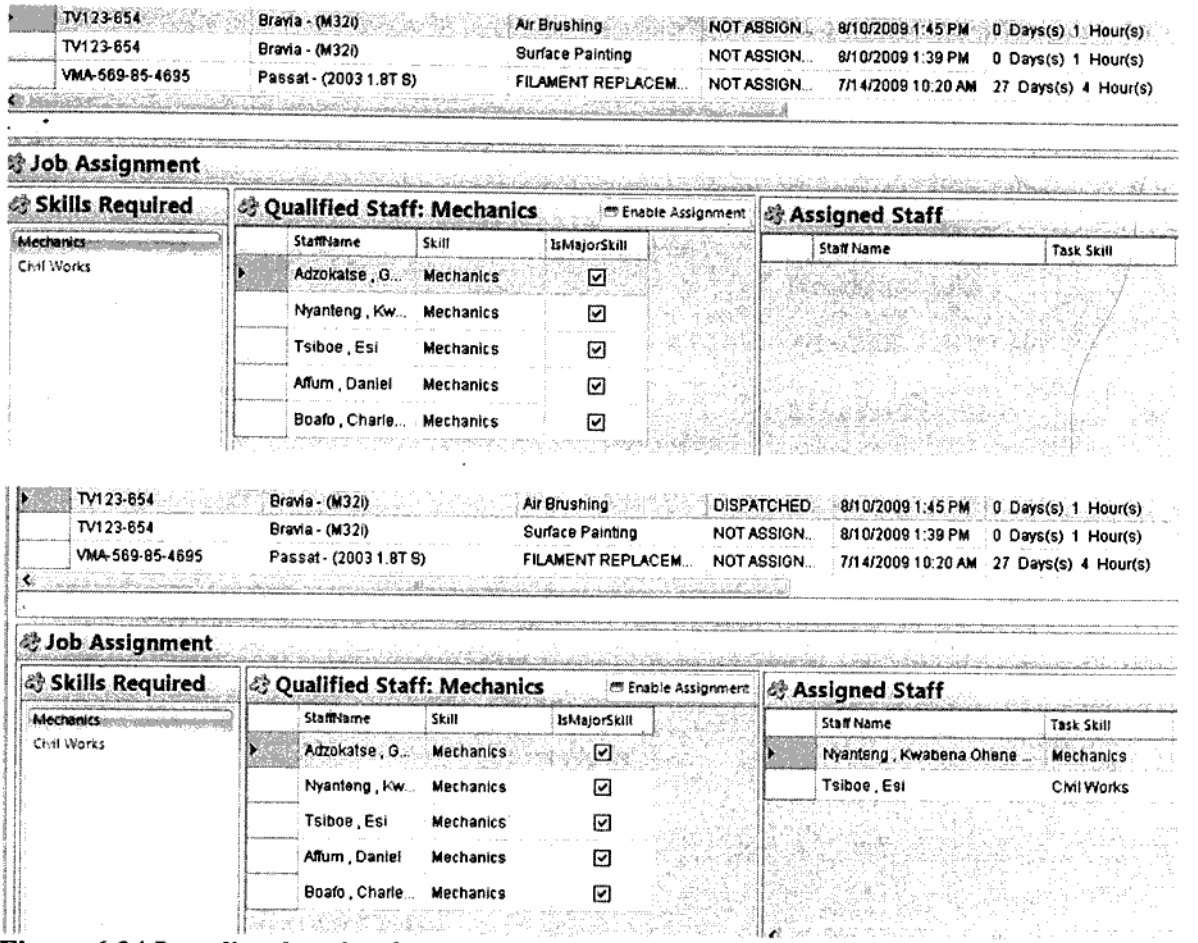


Figure 6.34 Item listed as due for service(top) , dispatched staff (bottom)

The staff assigned to the task are dispatched and issued out a dispatch form (Figure 6.35) and a work plan (Figure 6.36).

On completion, it is marked as completed (Figure 6.37) and placed in the maintenance history table (Figure 6.56.38).

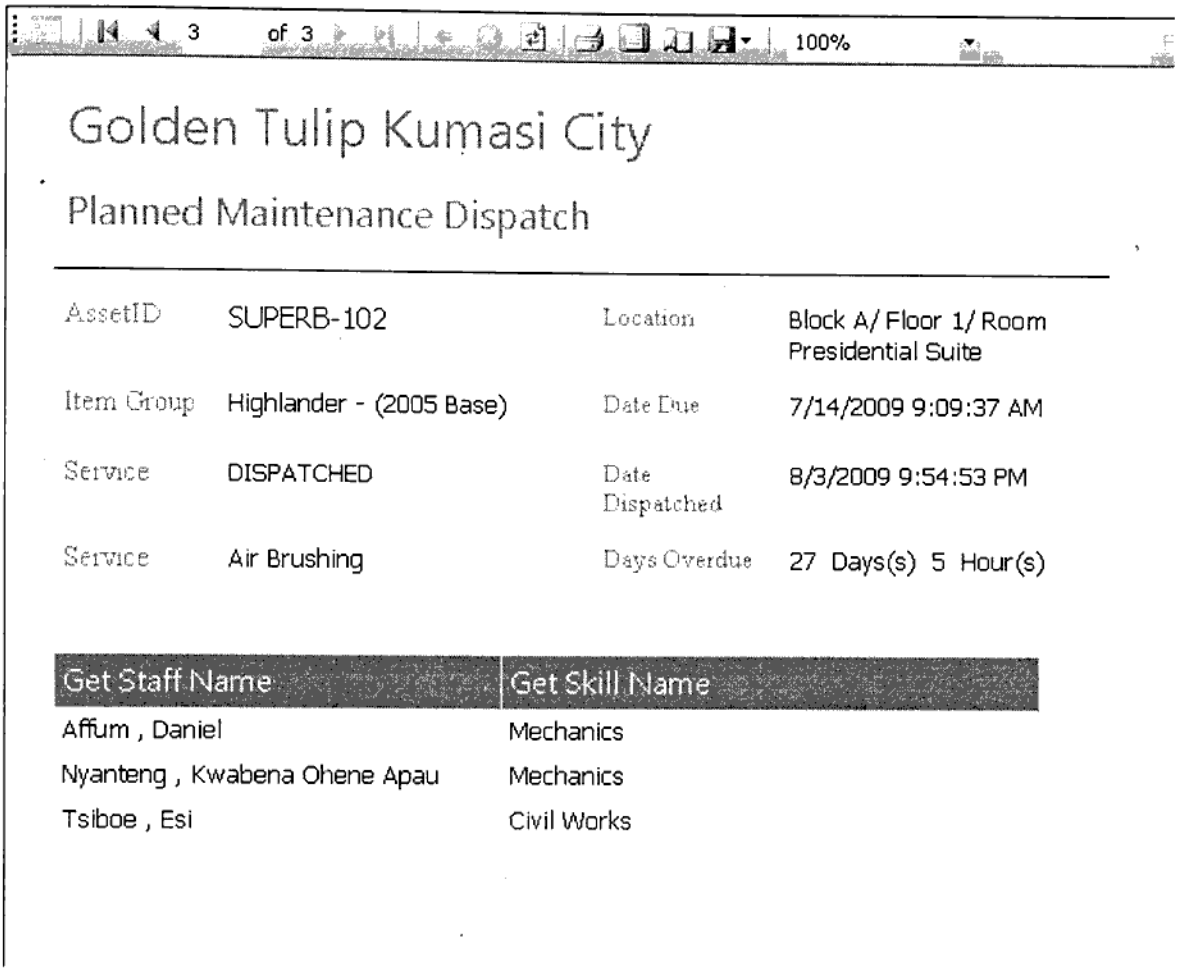


Figure 6.35 Sample Dispatch form

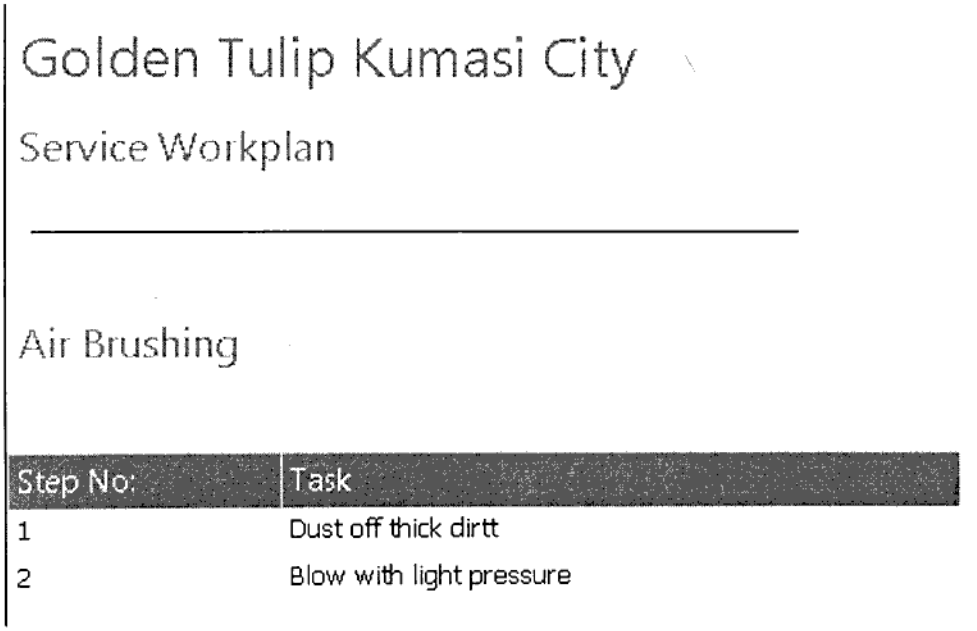


Figure 6.36 Service Work-plan

Golden Tulip Kumasi City

Asset Maintenance History

History ID	20	Date of Entry	8/10/2009 2:51:50 PM
Asset ID	TV123-654	Type	PLANNED
Type	Bravia - (M32i)	Cost	50
Fault/Service	Air Brushing	Date of Completion	8/10/2009 2:48:46 PM
Fault/Service	8/10/2009 2:44:00 PM		

SKill Name	Staff Assigned
Mechanics	Nyanteng , Kwabena Ohene Apau
Civil Works	Tsiboe , Esi

Figure 6.37 Report on completed planned task

Golden Tulip Kumasi City

Maintenance Summary

Asset ID	Type	Planned Count	UnPlanned Count	Total Service Count	Plannned Cost	Unplanned Cost	Total Cost
GTKC/KS/MAINT/2009/001	Rav4 - (2005 Limited)	4	1	5	1344	300	1644
HM90LKJT	Passat - (2003 1.8T S)	7		7	1974		1974
JNK-888-9658	Highlander - (2005 Base)			0			0
KLH-8796582	Highlander - (2005 Base)	2		2	445		445
SUPERB-102	Highlander - (2005 Base)			0			0
TEST-235-1253	Passat - (2003 1.8T S)			0			0
TST1105-1586	Passat - (2003 1.8T S)			0			0
TV123-654	Bravia - (M32i)	1	1	2	50	520	570
VMA-569-85-4695	Passat - (2003 1.8T S)			0			0
YTPWFG/986	Rav4 - (2005 Limited)			0			0

Figure 6.38 Maintenance Summary Statistics

7 CONCLUSIONS AND RECOMMENDATIONS

7.1 Conclusion

Maintaining a vast number of assets on a regular basis forms a huge challenge to many organisations especially, those in the hospitality sector. The GT Maintenance Management Software has addressed some of these challenges. It addresses asset locations, asset classification and categorisation.

It also addresses staff information, qualifications and work-shifts.

It enables users place unplanned requests and provides an interface for tracking the status of those requests.

Planned maintenance scheduling and staff suggestions are also tackled.

By taking care of some of these issues, the software will assist organisations within the hospitality industry to give better quality of service to their customers. It will also enable the avoidance of many expensive repairs due to adherence to the timely preventative maintenance practices.

7.2 Recommendations

Software management systems are designed to facilitate work. For the GTMMS to be successful, it requires the devotion of its user. The software can only prompt of an impending maintenance activity but it is up to the human factor to ensure that such suggestions are adhered to.

Manual Checks on the maintenance schedules of assets should, occasionally, be done to verify that the results of the system tallies with what pertains on the ground.

Problematic equipment should be considered for replacement to minimise servicing costs.

In the future, the Unplanned Maintenance Client should be ported to a web based interface to make it more accessible. This is because the current version of the Unplanned Maintenance Client is required to be installed on the requesting computer. A web based approach will eliminate the need for such an installation thereby making the Client accessible virtually on any computer connected to the network.

8 References

Ahmed, M., Garrett, C., & Faircloth, J. (2002). *ASP.NET Web Developer's Guide.* Sygress Publishing Inc.

Bryla, B. (2003). *Oracle 9i DBA Jump Start.* Sybex Inc.

Buzzle.com. (2008, 8 5). Retrieved July 20, 2009, from Advantages of Relational Databases: http://www.buzzle.com/articles/advantages-of-relational-databases.html

Clever Solutions. (2009). *Clever CMMS user friendly preventative maintenance software.* Retrieved 7 12, 2009, from Clever CMMS: The Maintenace Management Solution: http://www.clevercmms.com

CWORKS SYSTEMS. (2009). *CWorks: lowcost and free.* Retrieved 7 13, 2009, from CWorks CMMS: http://cworks.com.my

Dewson, R. (2006). *Beginning SQL Server 2005 for Developers.* Apress.

Encarta. (2009). Retrieved July 18, 2009, from http://encarta.msn.com: http://encarta.msn.com/dictionary_1861628101/management.html

IDCON. (2009). Retrieved July 15, 2009, from Reliability and Maintenance Definitions: http://www.idcon.com/maintenance-definition-M.htm

Kauffman, J., Claudio, F., Chinnathampi (Das), J., Greenwood, J., & Matsik, B. (2002). *ASP.Net Databases Using VB.NET.* Wrox Press Ltd.

McManus, J. P., & Goldstein, J. (2003). *Database Access with Visual Basic .NET.* Addison Wesley.

Petroutsos, E., & Bilgin, A. (2002). *Mastering Visula Basic .Net Database Programming.* Sybex Inc.

Price, J. (2003). *Mastering C# Database Programming.* Sybex INc.

(2005). Understanding n-tier Design. In B. Sempf, *Visual Basic 2005 for Dummies* (p. 73). Wiley Publishing Inc.

Shead, M. (2007, January 30). *http://www.leadership501.com.* Retrieved July 19, 2009, from Leadership501: http://www.leadership501.com/definition-of-management/21/

Smartware Group. (2008). *Bigfoot CMMS.* Retrieved 7 13, 2009, from Bigfoot Maintenace Software: http://bigfootcmms.com

Wikipedia. (2009, July 21). Retrieved July 23, 2009, from Management System: http://en.wikipedia.org/wiki/Management_system

Wikipedia. (2009, July 22). Retrieved July 22, 2009, from Computerized Maintenance Management System: http://en.wikipedia.org/wiki/CMMS

Wikipedia. (2009, July 22). Retrieved July 23, 2009, from Software Development Process: http://en.wikipedia.org/wiki/Software_development_process

Wikipedia. (2009, July 20). Retrieved July 23, 2009, from CodeGear Delphi: http://en.wikipedia.org/wiki/Borland_Delphi

Wikipedia. (2009, July 19). Retrieved July 19, 2009, from Visual Basic.Net: http://en.wikipedia.org/wiki/Visual_Basic.Net

Wikipedia.org. (2009, July 25). Retrieved July 22, 2009, from Programming Languages: http://en.wikipedia.org/wiki/Programming_languages

Wiktionary. (2009, July 18). Retrieved July 20, 2009, from Wiktionary.org: http://en.wiktionary.org/wiki/maintenance

Wkipedia. (2009, July 26). Retrieved July 27, 2009, from Database Normalization: http://en.wikipedia.org/wiki/Normalization_(database)

Appendices

Appendix A: Database Structure.

The following script is a description of the database structure

```
USE [master]
GO
/****** Object:  Database [GoldenData]    Script Date: 08/10/2009
15:05:42 ******/
IF NOT EXISTS (SELECT name FROM sys.databases WHERE name =
N'GoldenData')
BEGIN
CREATE DATABASE [GoldenData] ON  PRIMARY
( NAME = N'GoldenData', FILENAME = N'D:\My Thesis\GoldenData.mdf' ,
SIZE = 4096KB , MAXSIZE = UNLIMITED, FILEGROWTH = 1024KB )
 LOG ON
( NAME = N'GoldenData_log', FILENAME = N'D:\My
Thesis\GoldenData_log.LDF' , SIZE = 1024KB , MAXSIZE = UNLIMITED,
FILEGROWTH = 10%)
END

GO
EXEC dbo.sp_dbcmptlevel @dbname=N'GoldenData', @new_cmptlevel=90
GO
IF (1 = FULLTEXTSERVICEPROPERTY('IsFullTextInstalled'))
begin
EXEC [GoldenData].[dbo].[sp_fulltext_database] @action = 'enable'
end
GO
GO
ALTER DATABASE [GoldenData] SET DB_CHAINING OFF
USE [GoldenData]
GO
/****** Object:  Table [dbo].[Category]    Script Date: 08/10/2009
15:05:45 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[Category]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[Category](
      [CategoryID] [int] IDENTITY(1,1) NOT NULL,
      [CategoryName] [nvarchar](50) NOT NULL,
      [CategoryDescription] [nvarchar](500) NULL,
 CONSTRAINT [PK_Category] PRIMARY KEY CLUSTERED
(
      [CategoryID] ASC
```

```
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
END
GO

/****** Object:  Index [IX_Category]    Script Date: 08/10/2009
15:05:45 ******/
IF NOT EXISTS (SELECT * FROM sys.indexes WHERE object_id =
OBJECT_ID(N'[dbo].[Category]') AND name = N'IX_Category')
CREATE UNIQUE NONCLUSTERED INDEX [IX_Category] ON [dbo].[Category]
(
      [CategoryName] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[FormsInUse]    Script Date: 08/10/2009
15:05:45 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[FormsInUse]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[FormsInUse](
      [FormName] [nvarchar](50) NOT NULL,
      [InUse] [bit] NOT NULL,
      [UserName] [nvarchar](50) NULL,
      [TimeStarted] [datetime] NULL,
 CONSTRAINT [PK_FormsInUse] PRIMARY KEY CLUSTERED
(
      [FormName] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
END
GO
/****** Object:  Table [dbo].[Brand]    Script Date: 08/10/2009
15:05:45 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[Brand]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[Brand](
      [BrandCode] [int] IDENTITY(1,1) NOT NULL,
      [BrandName] [nvarchar](50) NOT NULL,
      [Description] [nvarchar](500) NULL,
 CONSTRAINT [PK_Brand] PRIMARY KEY CLUSTERED
(
      [BrandCode] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
END
GO
```

```
/****** Object:  Index [IX_Brand]    Script Date: 08/10/2009 15:05:45
******/
IF NOT EXISTS (SELECT * FROM sys.indexes WHERE object_id =
OBJECT_ID(N'[dbo].[Brand]') AND name = N'IX_Brand')
CREATE UNIQUE NONCLUSTERED INDEX [IX_Brand] ON [dbo].[Brand]
(
      [BrandName] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Shift]    Script Date: 08/10/2009
15:05:45 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[Shift]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[Shift](
      [ShiftCode] [nvarchar](50) NOT NULL,
      [ShiftDescription] [nvarchar](500) NULL,
      [BeginTime] [datetime] NOT NULL,
      [EndTime] [datetime] NOT NULL,
      [AutoID] [int] IDENTITY(1,1) NOT NULL,
      [ComputedBeginTime]  AS (right([BeginTime],(7))),
      [ComputedEndTime]  AS (right([EndTime],(7))),
 CONSTRAINT [PK_Shift] PRIMARY KEY CLUSTERED
(
      [ShiftCode] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
END
GO

/****** Object:  Index [IX_Shift]    Script Date: 08/10/2009 15:05:45
******/
IF NOT EXISTS (SELECT * FROM sys.indexes WHERE object_id =
OBJECT_ID(N'[dbo].[Shift]') AND name = N'IX_Shift')
CREATE NONCLUSTERED INDEX [IX_Shift] ON [dbo].[Shift]
(
      [BeginTime] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
GO

/****** Object:  Trigger [dbo].[DeleteShift]    Script Date: 08/10/2009
15:05:45 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

IF NOT EXISTS (SELECT * FROM sys.triggers WHERE object_id =
OBJECT_ID(N'[dbo].[DeleteShift]'))
EXEC dbo.sp_executesql @statement = N'CREATE TRIGGER
[dbo].[DeleteShift] on [dbo].[Shift]
for delete
```

```
as
declare @CodeDeleted nvarchar(50)

select @CodeDeleted = ShiftCode from deleted

update StaffShift set StaffShift.MondayShiftCode = ''FREE'' where
StaffShift.MondayShiftCode = @CodeDeleted
update StaffShift set StaffShift.TuesdayShiftCode = ''FREE'' where
StaffShift.TuesdayShiftCode = @CodeDeleted
update StaffShift set StaffShift.WednesdayShiftCode = ''FREE'' where
StaffShift.WednesdayShiftCode = @CodeDeleted
update StaffShift set StaffShift.ThursdayShiftCode = ''FREE'' where
StaffShift.ThursdayShiftCode = @CodeDeleted
update StaffShift set StaffShift.FridayShiftCode = ''FREE'' where
StaffShift.FridayShiftCode = @CodeDeleted
update StaffShift set StaffShift.SaturdayShiftCode = ''FREE'' where
StaffShift.SaturdayShiftCode = @CodeDeleted
update StaffShift set StaffShift.SundayShiftCode = ''FREE'' where
StaffShift.SundayShiftCode = @CodeDeleted
'
GO

/****** Object:  Trigger [dbo].[Updatehift]    Script Date: 08/10/2009
15:05:45 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

IF NOT EXISTS (SELECT * FROM sys.triggers WHERE object_id =
OBJECT_ID(N'[dbo].[Updatehift]'))
EXEC dbo.sp_executesql @statement = N'CREATE TRIGGER [dbo].[Updatehift]
on [dbo].[Shift]
for update

as
declare @CodeDeleted nvarchar(50), @NewCode nvarchar(50)

select @CodeDeleted = ShiftCode from deleted
select @NewCode = ShiftCode from inserted


update StaffShift set StaffShift.SundayShiftCode = @NewCode where
StaffShift.SundayShiftCode = @CodeDeleted
'
GO
IF NOT EXISTS (SELECT * FROM
::fn_listextendedproperty(N'MS_Description' , N'SCHEMA',N'dbo',
N'TABLE',N'Shift', N'COLUMN',N'ComputedBeginTime'))
EXEC sys.sp_addextendedproperty @name=N'MS_Description',
@value=N'Contains the text representtion of the extracted commencement
time' , @level0type=N'SCHEMA',@level0name=N'dbo',
@level1type=N'TABLE',@level1name=N'Shift',
@level2type=N'COLUMN',@level2name=N'ComputedBeginTime'
GO
```

```
/****** Object:  Table [dbo].[StoresTransactionTable]    Script Date:
08/10/2009 15:05:46 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[StoresTransactionTable]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[StoresTransactionTable](
      [TransactionID] [int] IDENTITY(1,1) NOT NULL,
      [TransactionDate] [datetime] NOT NULL,
      [ItemCode] [nvarchar](50) NOT NULL,
      [ItemName] [nvarchar](50) NOT NULL,
      [ItemCategoryName] [nvarchar](50) NULL,
      [ItemCategoryCode] [nvarchar](50) NULL,
      [InitialQuantiyAvailable] [int] NULL,
      [QuantityTaken] [int] NULL,
      [QuantityRemaining] [int] NULL,
      [NameOfSupplier] [nvarchar](50) NULL,
      [NameOfRequester] [nvarchar](50) NULL,
 CONSTRAINT [PK_StoresTransactionTable] PRIMARY KEY CLUSTERED
(
      [TransactionID] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
END
GO
/****** Object:  Table [dbo].[Skill]    Script Date: 08/10/2009
15:05:46 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[Skill]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[Skill](
      [SkillID] [int] IDENTITY(1,1) NOT NULL,
      [SkillName] [nvarchar](50) NOT NULL,
      [SkillDescription] [nvarchar](500) NULL,
 CONSTRAINT [PK_Skill] PRIMARY KEY CLUSTERED
(
      [SkillID] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
END
GO
/****** Object:  UserDefinedFunction [dbo].[TimeOnly]    Script Date:
08/10/2009 15:05:46 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[TimeOnly]') AND type in (N'FN', N'IF', N'TF', N'FS',
N'FT'))
```

```
BEGIN
/****** Object:  Table [dbo].[WorkStatus]    Script Date: 08/10/2009
15:05:46 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[WorkStatus]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[WorkStatus](
	[AutoID] [int] IDENTITY(1,1) NOT NULL,
	[SenderID] [nvarchar](50) NULL,
	[ReceipientID] [nvarchar](50) NULL,
	[Title] [nvarchar](100) NULL,
	[MessageBody] [nvarchar](800) NULL,
	[DateSent] [datetime] NULL,
	[MessageType] [nvarchar](50) NULL,
	[SenderName] [nvarchar](50) NULL,
	[ReceipientName] [nvarchar](50) NULL,
 CONSTRAINT [PK_WorkStatus] PRIMARY KEY CLUSTERED
(
	[AutoID] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
END
GO
/****** Object:  Table [dbo].[CategoryTable]    Script Date: 08/10/2009
15:05:46 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id
OBJECT_ID(N'[dbo].[CategoryTable]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[CategoryTable](
	[CategoryCode] [int] IDENTITY(1,1) NOT NULL,
	[CategoryName] [nvarchar](50) NOT NULL,
 CONSTRAINT [PK_CategoryTable] PRIMARY KEY CLUSTERED
(
	[CategoryCode] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
END
GO
IF NOT EXISTS (SELECT * FROM
::fn_listextendedproperty(N'MS_Description' , N'SCHEMA',N'dbo',
N'TABLE',N'CategoryTable', N'COLUMN',N'CategoryCode'))
EXEC sys.sp_addextendedproperty @name=N'MS_Description', @value=N'An
auto generated index' , @level0type=N'SCHEMA',@level0name=N'dbo',
@level1type=N'TABLE',@level1name=N'CategoryTable',
@level2type=N'COLUMN',@level2name=N'CategoryCode'
GO
/****** Object:  Table [dbo].[MaintenanceHistory]    Script Date:
08/10/2009 15:05:46 ******/
SET ANSI_NULLS ON
```

```
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[MaintenanceHistory]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[MaintenanceHistory](
	[HistoryID] [int] IDENTITY(1,1) NOT NULL,
	[DateOfEntry] [datetime] NULL,
	[ItemClassName] [nvarchar](50) NULL,
	[AssetID] [nvarchar](50) NOT NULL,
	[ServiceID] [nvarchar](50) NULL,
	[ServiceOrFaultName] [nvarchar](50) NOT NULL,
	[DateWhenDue] [datetime] NULL,
	[DateCompleted] [datetime] NULL,
	[HoursOverDue] [int] NULL,
	[MaintenanceCost] [float] NULL,
	[PlannedOrUnplanned] [nvarchar](50) NULL,
	[DateStaffWasAssignedWork] [datetime] NULL,
	[HoursStaffOverDue] [int] NULL,
	[AssetName] [nvarchar](50) NULL,
	[ScheduleID] [nvarchar](50) NULL,
	[RequestID] [int] NULL,
 CONSTRAINT [PK_MaintenanceHistory] PRIMARY KEY CLUSTERED
(
	[HistoryID] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
END
GO
/****** Object:  Table [dbo].[StoresTable]    Script Date: 08/10/2009
15:05:46 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[StoresTable]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[StoresTable](
	[ItemCode] [nvarchar](50) NOT NULL,
	[ItemName] [nvarchar](50) NULL,
	[CategoryCode] [nvarchar](50) NULL,
	[Location] [nvarchar](50) NULL,
	[QuantityRemaining] [int] NULL CONSTRAINT
[DF_StoresTable_QuantiyRemaining]  DEFAULT ((0)),
	[RestockLevel] [int] NULL CONSTRAINT
[DF_StoresTable_RestockLevel]  DEFAULT ((0)),
	[DateOfLastRestock] [datetime] NULL CONSTRAINT
[DF_StoresTable_DateOfLastRestock]  DEFAULT (getdate()),
	[QuantityOfLastRestock] [int] NULL CONSTRAINT
[DF_StoresTable_QuantityOfLastRestock]  DEFAULT ((0)),
	[Description] [nvarchar](100) NULL,
	[AutoID] [int] IDENTITY(1,1) NOT NULL,
 CONSTRAINT [PK_StoresTable] PRIMARY KEY CLUSTERED
(
	[ItemCode] ASC
```

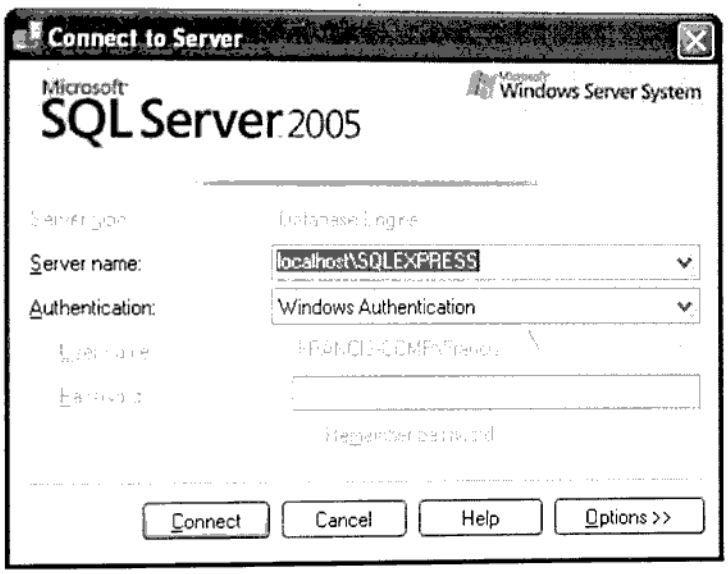
```
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF  ON [PRIMARY]
) ON [PRIMARY]
END
GO

/****** Object:  Index [IX_StoresTable]    Script Date: 08/10/2009
15:05:46 ******/
IF NOT EXISTS (SELECT * FROM sys.indexes WHERE object_id =
OBJECT_ID(N'[dbo].[StoresTable]') AND name = N'IX_StoresTable')
CREATE UNIQUE NONCLUSTERED INDEX [IX_StoresTable] ON
[dbo].[StoresTable]
(
      [ItemName] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF  ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Fault]    Script Date: 08/10/2009
15:05:46 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[Fault]') AND type in (N'U'))
BEGIN
CREATE TABLE [dbo].[Fault](
      [FaultID] [int] IDENTITY(1,1) NOT NULL,
      [FaultName] [nvarchar](50) NULL,
      [FaultDescription] [nvarchar](100) NULL,
 CONSTRAINT [PK_Fault] PRIMARY KEY CLUSTERED
(
      [FaultID] ASC
)WITH (PAD_INDEX  = OFF, IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
END
GO
```

Appendix C: Sample Code

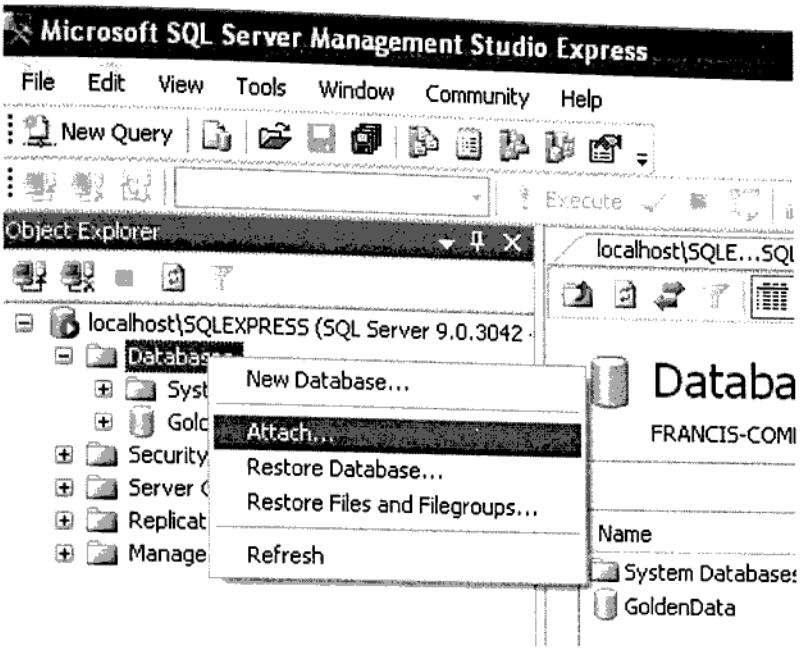
```
Public Function Update(ByVal updateString As String) As Boolean
        Dim querry As String
        Dim con As New SqlConnection()
        con.ConnectionString = Module1.Connection
        Try

            querry = updateString
            Dim cmd As New SqlCommand(querry, con)
            con.Open()
            cmd.ExecuteNonQuery()

            cmd.Dispose()
            con.Close()
            'success = True
            Return True
        Catch ex As SqlException
            If ex.Number = 2627 Or ex.Number = 2601 Then
                MessageBox.Show("The record you are trying to add
already exists", "An Error has occured", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            Else
                MessageBox.Show(ex.Message + ex.Number.ToString(), "An
Error has occured", MessageBoxButtons.OK, MessageBoxIcon.Error)
            End If
            If con.State = ConnectionState.Open Then
                con.Close()
            End If
            Return False
        End Try


    End Function
```

Appendix D: Installation of GTMMS

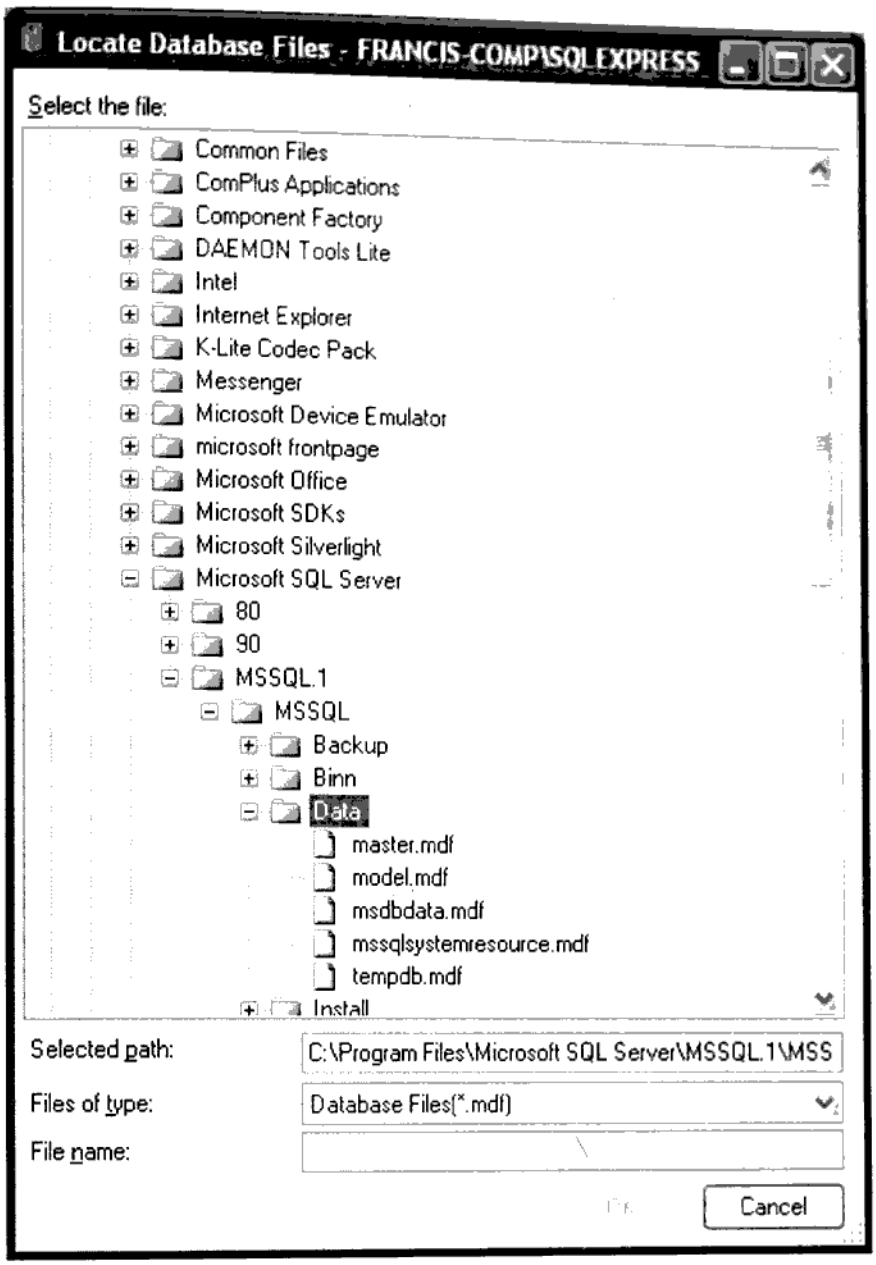
1. Locate Microsoft SQL Server Express Edition (SQLServer2005_SSMSEE.msi) from the Software\MManager\sqlexpress folder on the installation CD-ROM
2. Install Microsoft SQL Server Express Edition using the default settings
3. Locate Microsoft SQL Server Management Studio Express (SQLServer2005_SSMSEE.msi) from the Data folder on the installation CD-ROM
4. Install Microsoft SQL Server Management Studio Express edition
5. Start Microsoft SQL Server Management Studio Express



6. Click Connect button

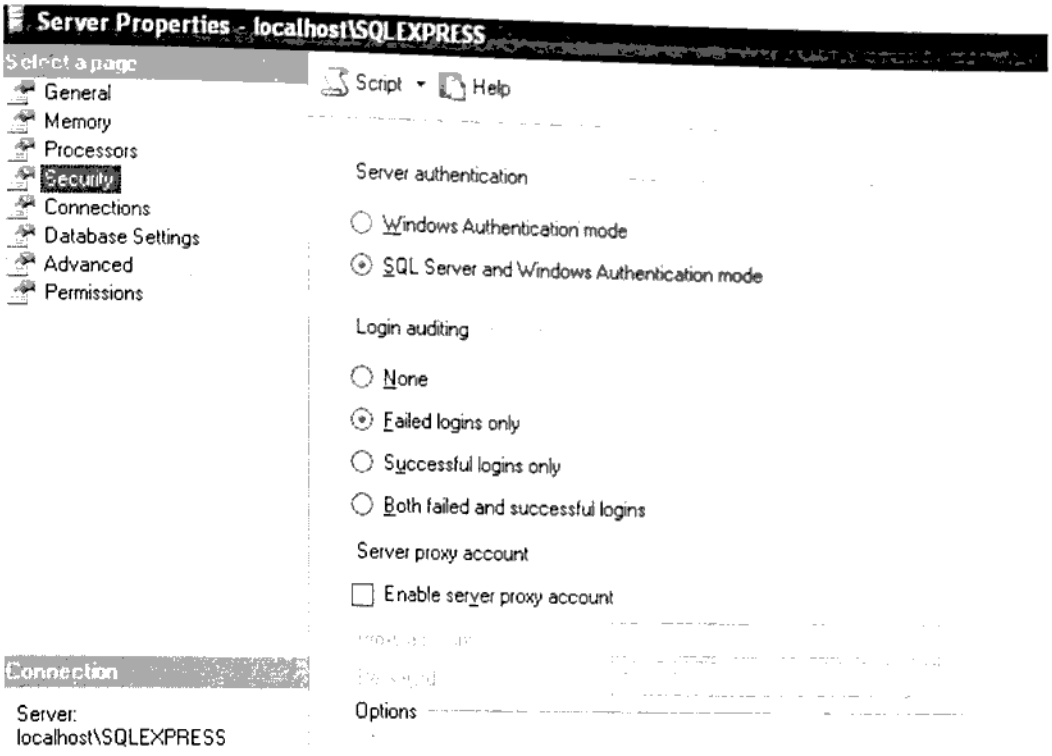


7. Right click the Database Folder and Select attach



8. Navigate to the location of the database and click OK

10. Right click the server icon and select properties



11. Select the Security option

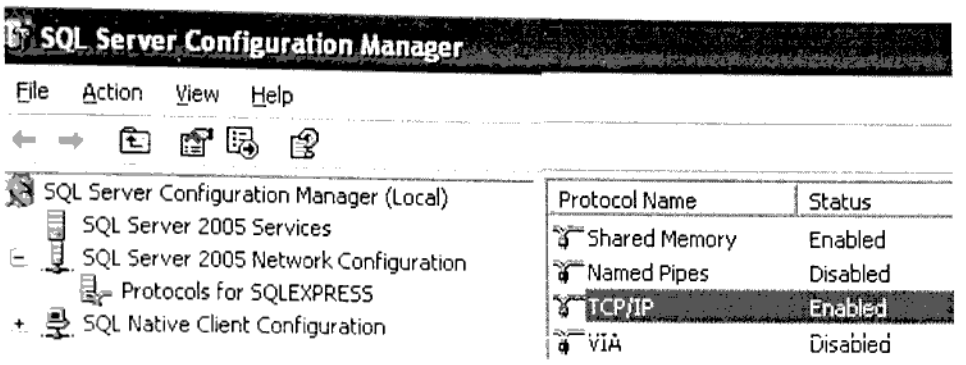
12. Under "Server Authentication'" select "SQL Server and Windows Authentication"

13. Click Ok

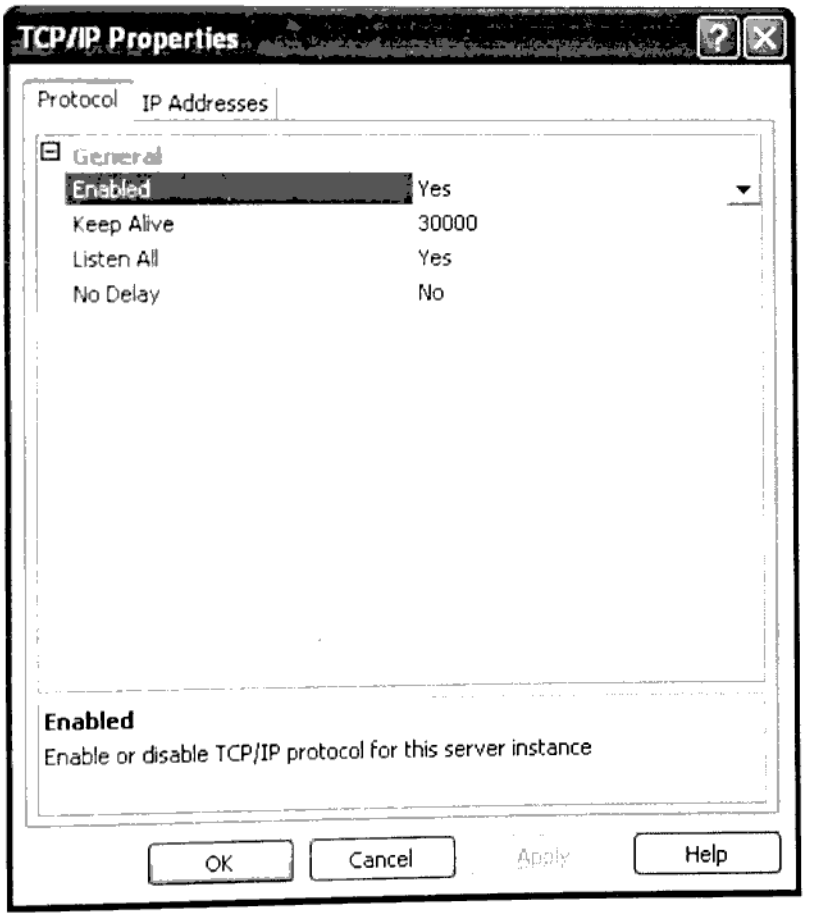
14. Navigate through the Windows Start Icon > All Programs > Microsoft SQL Server 2005 and select SQL Server Configuration Manager

15. Select the SQL Server Network Configuration Node > Protocols for SQLExpress

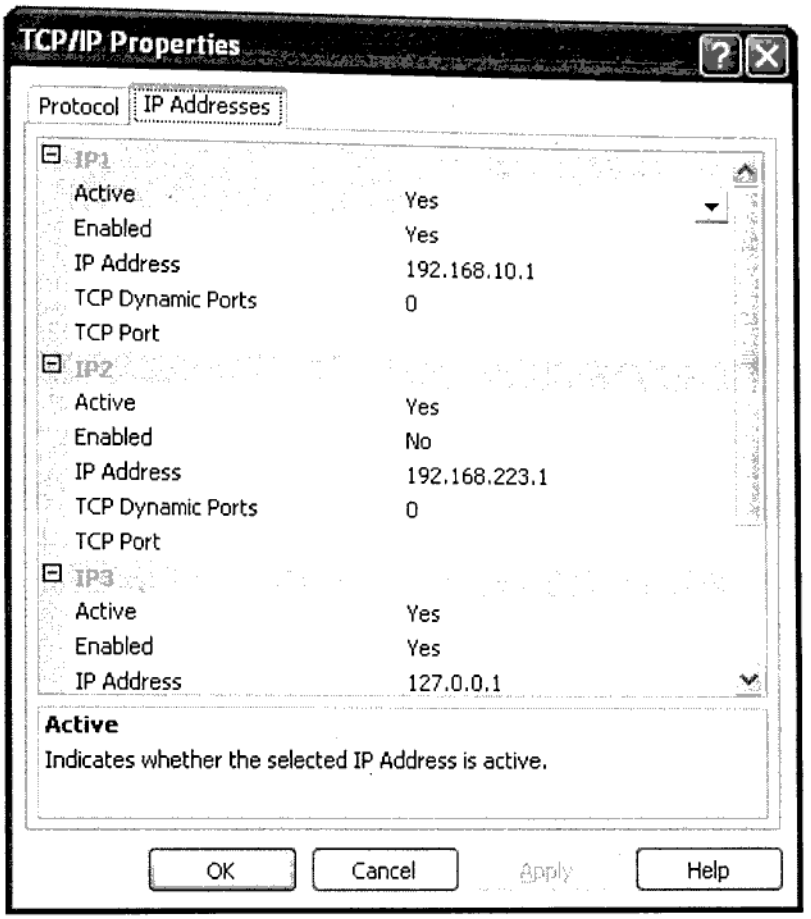
16. Right Click the TCP/IP option and select properties



17. On the Protocol tab, set the enabled option to Yes



18. On the IP Addresses tab, enable the IP address of the network interface card of your computer and set the 'Active' status to 'Yes'



19. Click OK and restart the server

20. Locate the LOGINGENERTOR.exe supplied on the CD from Software\Login\Debug and click the Generate Login button

21. Close the login generator.

22. Locate the setup.exe file supplied on the CD from Software\MManager and install using the default settings

23. For each client that needs to be installed , Locate the setup.exe file supplied on the CD from Software\Clientand install using the default settings

24. Start the GTMMS Maintenance Manager.

25. Use default username 'admin' and password 'gold' to login

Appendix D: GTMMS Usage Manual

For the manual on the usage of the GTMMS, locate the GettingStarted.html file from the Manual folder on the installation CD-ROM