Computerized System for Containerized Cargo Handling – Feeder Service

By

P.A.C. Deepani

01/AS/005

This thesis is submitted in partial fulfillment of the requirements for the degree of

Bachelor of sciences (Applied sciences)

In

Physical Sciences

Department of Physical sciences

Faculty of Applied sciences

Sabaragamuwa University of Sri Lanka

Buttala

June 2005

Declaration

I certify that this dissertation does not incorporate without acknowledgment of any material previously submitted for a degree or diploma in any other university and to the best of my knowledge and belief this does not contain any material previously published in writing or orally communicated by another person where due reference is made in the text.

[illegible]	20/08/2005
P.A.C.Deepani	Date

Certified by

External Supervisor
Mr.M.C.P.K.Wimalasena
IT Manager
Green Lanka Shipping Ltd
46/46, Nawam Mawatha
Colombo 02
Tel: 011 2302100

GREENLANKA SHIPPING LTD.
GREENLANKA TOWERS
46/46, NAWAM MAWATHA,
COLOMBO 02.

Signature

20/08/2005
Date

Internal Supervisor
Mr.Jayalath Ekanayake
Senior Lecturer
Faculty of Applied Sciences
Sabaragamuwa University of Sri Lanka
Buttala
Tel: 055 2273986

[illegible]
Signature

30/08/2005
Date

Department Head
Dr. (Mrs) M.N.Wickramaratne
Head/Dept. of Physical Sciences
Faculty of Applied Sciences
Sabaragamuwa University of Sri Lanka
Buttala
Tel: 055 2273986

[illegible]
Signature

30/08/2005
Date

DR. (MRS.) M. N. WICKRAMARATNE
Acting Head/Dept. of Physical Sciences
Faculty of Applied Sciences
Sabaragamuwa University of Sri Lanka
Buttala.

ACKNOWLEDGMENT

It is a great pleasure for me to acknowledge assistance and contributions of personal to this effort.

First and foremost I would like to thank the management of Green Lanka Shipping Ltd for giving me the opportunity to carry out this project. I express my sincere gratitude to my internal supervisor, Mr. Jajalath Ekanayake who gave me the guidance and encouragement to under take this project and always willing to spare his time to render me with valuable advises. I would also like to express my gratitude to Mr. M.D.S.L.P.Gunathilaka and Mr. M.C.P.K.Wimalasena for encouraging me from their guidance by appointed as my external supervisors in this effort.

My sincere appreciation is expressed to Dr.D.B.M.Wickramaratne, Dean of the faculty and Dr (Mrs) Nirmali Wickramaratne, Head of the department for guiding me towards a successful completion of my degree.

A special word of thank is due to Mr. Suneth, staff members and the Management of Sea Services (PVT) Ltd, who were always willing to help me through out the system analysis phase.

Also I would like to extend my thanks to all the staff members in IT section of Green Lanka Shipping Ltd, for their great assistance they gave me whenever it is needed.

Abstract

Feeder Service was the main service provided by Sea Services (PVT) Ltd, which is a computerized system used to capture the details of containers coming to Colombo and going out of Colombo, and report them to Sri Lanka Port Authority (SLPA). The existing system was developed using CLIPER language and database in dBase. The system was not user friendly and it was not a multi-user system. Entering TDR details and converting them into a standard format, which is called Port Transfer Format, was done manually, hence it was a time consuming process. Moreover, data retrieval and transaction update processes were slow. The system did not support inquiry and marketing statistics and also did not generate Management Information Statistics (MIS) reports. In addition to these, the system was not secure and reliable.

The overall objective of the study was to develop a user-friendly Windows base system and introduce a client server network solution and overcome the above drawbacks.

Therefore the system was developed using the waterfall model. According to the model, analysis was carried out by studying the existing system, having discussions with the management, interviewing the potential users of the system, collecting the documents and collecting the reports generated by the system. In the design stage, relevant data flow diagrams were designed to the structured method and level them up to 3rd. The database was designed using relational model and the normalization was done up to 3rd normal from. Furthermore, the user interfaces were designed using VB 6.0. and the database was implemented using SQL serve 2000. The ActiveX Data Objects (ADO) method was used to access the database through user interfaces.

The outcome of the software project was a user-friendly Windows base system to handle the information related to Feeder Service of the Sea Services (PVT) Ltd. The system met all the user requirements except the MIS reports. The functions of generating MIS reports and Inquiry were not implemented due to time constraints. The management of the Green Lanka Shipping Ltd accepted the system; introducing online accessible features and report generating tools to the system could do further enhancements.

Contents

List of Figures

Chapter 01

1. INTRODUCTION

1.1 GREEN LANKA Shipping Ltd

1.1.1 Background

GREEN LANKA Shipping Ltd is a leading shipping company in Sri Lanka, and it is the dedicated agency for EVERGREEN Marine Corporation group of Taiwan (Evergreen International USA / Green Compass Marine SA PANAMA) and HATSU Marine Limited to represent Evergreen Group Vessels since 1989. Over 300 vessels are handled by company an account of EVERGREEN Marine Corporation and HATSU Marine Limited at the port of Colombo. The company was incorporated in 1989, and it is expanding the reach with 14 outstanding companies, with the workforce of around 250 employees. The 14 companies bound together are,

- Green Lanka Shipping Ltd
- Sea Services (PVT) Ltd
- TRISTINO Lanka Shipping Limited
- Unicorns Cleaning & Forwarding (PVT) Ltd
- Green Lanka Travels & Tourism (PVT) Ltd
- GNKP Cruise Services (PVT) Ltd
- Green Lanka Property Developer (PVT) Ltd
- Allied Lanka Travels(PVT)Ltd
- Green Lanka Container Freight Station Terminal(PVT)Ltd
- Ocean Lanka Services(PVT)Ltd
- Unicorn Overseas Services(PVT)Ltd
- Techno Fiber Lanka(PVT)Ltd
- Global Express Freight(PVT)Ltd
- Green Pick(PVT)Ltd

Sea Services (PVT) Ltd is a one of leading companies of the Green Lanka Group. It was incorporated in 1994 to cater to the feedering of cargo to the Indian sub continent from Colombo. At the inception of the Evergreen service to Colombo, they operated their dedicated feeder service and later on the feedering operation was handed over to their in house feedering company. The Sea Services (PVT) Ltd was the only Sri Lanka vessel owning Feeder Company operating in Colombo when it was formed in 1994. Since then, the company has flourished to become the number one feeder operator, playing between Colombo and Indian Port of Mumbai, Nava Shiva, Kalkata...etc.

1.2 Project Overview

1.2.1 Background

The Feeder Service System was a computerized system, which has been used to capture the details of containers coming to Colombo and going out of Colombo, (mainly from and to Indian sub continent ports) and report them to Sri Lanka Port Authority (SLPA). The existing feeder service system was developed in 1994, using CLIPER language and database in dBase, and currently used in Sea Services (PVT) Ltd.

The four main functions involve to the system are Terminal Departure Report (TDR), Discharging, Loading and Transshipment. And also Invoicing takes place at the end of the process. The system contains several modules, which are used by Operation dept., Documentation dept. and Accounting dept., do not have a proper link and organization. The invoicing module has been used for billing purposes for the containers, handled by the Sea Services (PVT) Ltd.

First of all the vessel schedule is updated for a new vessel or for each voyage. Vessels details contain code of the vessel, Name, Company and a record of whether it is main or feeder vessel. Vessel schedule contains vessel code, owner, inward and outward details such as voyage, Expected Time for Arrival (E.T.A), Expected Time for Departure (E.T.D.), Exchange rates and port references.

In the Discharging process, Load port sends a TDR to the company. A TDR contains serial Number (SI), Container No, Type/ Size, Line code, Gross weight in MT (Gr. Wt), Location of the container, Port of Discharge (POD), States, Vessel code & Voyage code. A port has a code that Sea Services uses, Name, Country and SLPA code which refers to the same port that SLPA uses and also UN code which is the internationally accepted way of representing a port.

Then the TDR is manually entered to the system. The agents send declaration forms. They are used to confirm the TDR entries.

Main Line Operators are the agents for the containers. They have MLO code, Name and Address.

After confirming the TDR, all entries are taken into a standard format to send the TDR details to the SLPA, which is called Port Transfer Format. Then the data are transferred to the port (SLPA). Finally the system generates several reports.

1. Inward Reports
    - 1.1. Inward Declaration } send to SLPA
    - 1.2. Inward Summary } send to SLPA
    - 1.3. Cargo Manifest
    - 1.4. Delivery Order - send to Local Agent

In the Loading process agents send declarations with the details of goads which were loaded from Colombo. Those details are entered to the system and a TDR is generated with the TDR summary. That TDR is then sent to the discharging port (POD). The reports generating here are,

2. Outward Reports
    - 2.1 Outward Declaration } to SLPA
    - 2.2 Outward summary } to SLPA
    - 2.3 Bill of Lading – to Agent
    - 2.4 TDR } to POD
    - 2.5 TDR summary } to POD

Inward and outward reports are used to see the local discharging and local loading from Colombo. They don't contain the transshipment details. To see the transshipment details transshipment reports are generated and send them to SLPA.

3. Transshipment Reports
    - 3.1 Transshipment Container & Other Transshipment Cargo Handling Declaration- Inward
    - 3.2 T/ S Container & Other T/ S Cargo Handling Declaration Summary - Inward
    - 3.3 Transshipment Container & Other Transshipment Cargo Handling Declaration - Outward

3.4 T/ S Container & Other T / S Cargo Handling Declaration Summary – Outward

Finally system generates invoices and sends them to Local Agents. Then Receipts, Credits Notes and Outstanding Reports are generated.

1.2.2 Problem Identification

There were number of drawbacks in the existing system. The system did not support multi-user and also not user friendly, because the DB File system was outdated and the system was running on DOS. TDR details were entered manually. Since the TDR is a large document and vessels come frequently, entering the TDR details manually was a time consuming job. Further, the TDR had to be converted into a standard format to send the TDR details to SLPA, which is called Port Transfer Format. In that format, for every operator code the billing code appears as SEA, means that the SLPA will bill to Sea Services (PVT) Ltd for all the containers. Since most of the operator lines have direct billing from the SLPA, the billing codes were replaced manually. For that purpose, each and every operator code should be checked to change the billing code, so the job was further complicated and also time consuming. Moreover data retrieval and transaction update processes were also slow. The system did not support Inquiry and Marketing Statistics and also did not generate Management Information Statistics (MIS) reports. Also the system was not secure and reliable.

1.2.3 Overall Objective

Design a Feeder Service System to overcome the above drawbacks by introducing a Windows base user-friendly system using Graphical User Interface (GUI) with a shared database and a Client Server Network solution.

Chapter 02

2. LITERATURE REVIEW

2.1 The Software Process

The software process is the set of activities and associated results, which produce a software product.

There are four fundamental process activities, which are common to all software processes. The activities are:

I. Software specification – The functionality of the software and constrain on its operation must be defined
II. Software development – The software to meet the specification must be produced
III. Software validation – The software must be validated to ensure that it does what the customer wants
IV. Software Evolution – The software must evolve to meet changing customer needs.

Different software processes decompose these activities in different ways. The timing of the activities varies as does the results of each activity. According to the software process the way of reaching the end product will vary. The model will be selected according to the environment of customer requirement. If the wrong process is used, this will probably reduce the quality or the usefulness of the software product to be developed. There are a number of different general model of software development:

- The waterfall approach (Liner sequential model)
- Evolutionary development
- Formal Transformation
- System Assembly from reusable components

There are no clear arguments about quality and reliability of those models. They rely upon the user requirements.

2.1.1 The waterfall model

The first explicit model of the software development process was derived from other engineering processes. This takes the above activities and represents them as separate process phases such as requirement specification, software design, implementation, testing and so on. After each stage is defined it is 'signed – off' and development goes on to the following stage. Because of the cascade from one phase to another, this model is known as the 'waterfall model'. The waterfall model is shown below. (Sommerville, 2000)

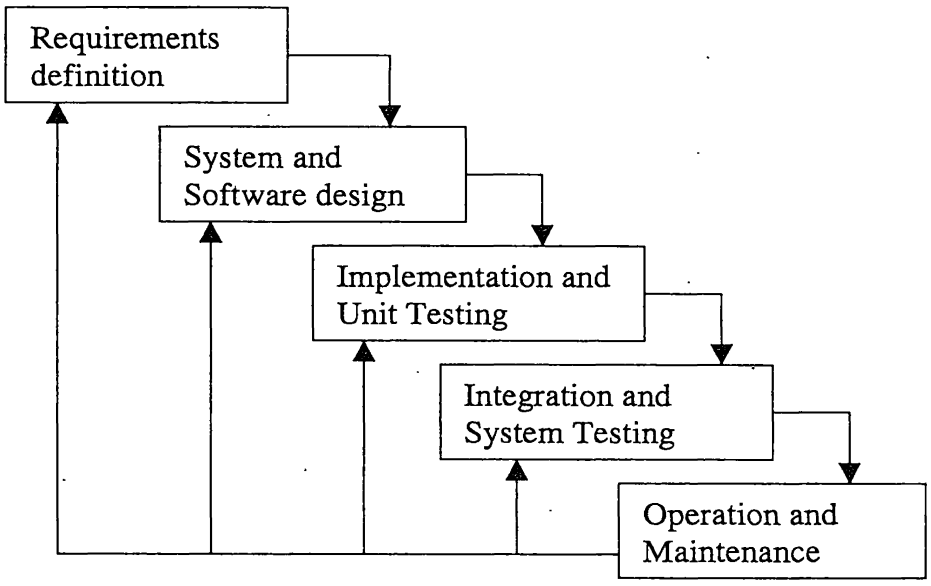

Requirements definition
System and Software design
Implementation and Unit Testing
Integration and System Testing
Operation and Maintenance

Figure 2.1. The Waterfall model (Source - Sommerville, 2000)

For this model nearly all requirements need to be understood and well defined.

2.1.2 Evolutionary development

Evolutionary development is based on the idea of development an initial implementation, exposing this to user comment and refining this through many versions until an adequate system has been developed.

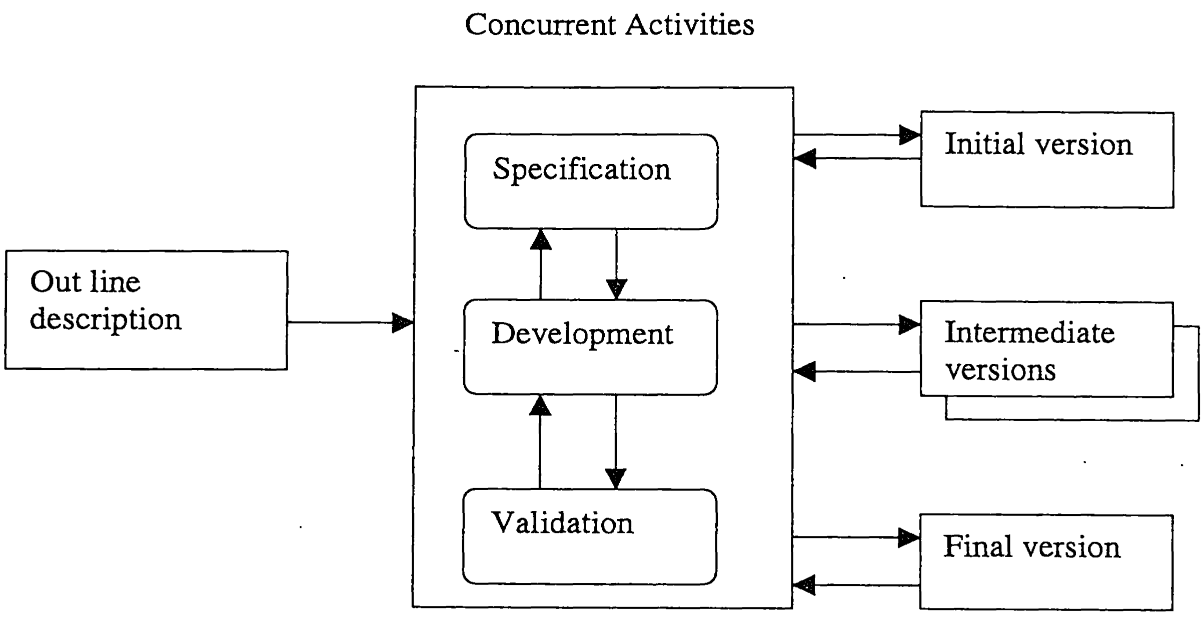


Figure 2.2. Evolutionary development (Source - Sommerville, 2000)

Rather than have separate specification, development and validation activities, there are carried out concurrently with rapid feedback across these activities.

There are two types of evolutionary development:

I. Exploratory Programming - work with the customer to explore their requirements and deliver a final system. The development starts with the parts of the system, which are understood. The system evolves by new features as they are proposed.

II. Throw-away Prototyping - objective of the process is to understand the customer's requirements and hence develop a better requirements definition for the system. The prototype concentrates an experimenting with those parts of the customer requirement which are poorly understood.

(Sommerville, 2000)

2.1.3 Spiral model

Each loop in the spiral represents a phase (as defined by management) of the software process.

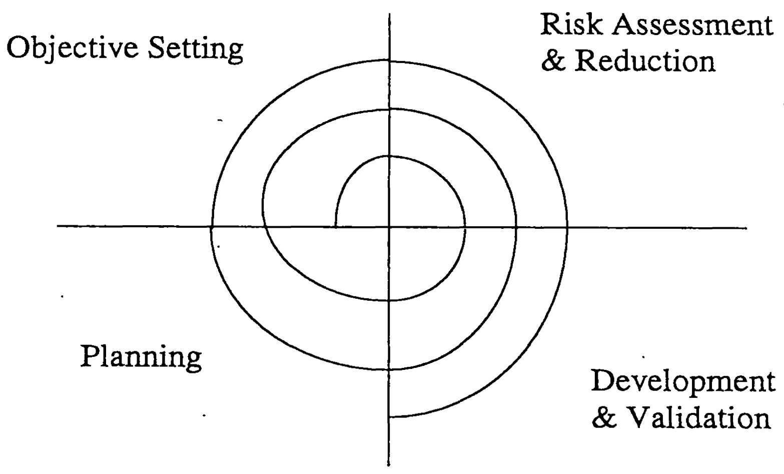


Figure 2.3. Spiral model

The spiral model is a Meta model, because it can incorporate any other model in it. Whether one or the other model is chosen depends on the level of risk. This model focuses on systems that are more problematic, or have a higher risk factor, than others. Another important factor is that the model is non linear.

2.2 Software Development Life Cycle

The development process undergoes several stages. These stages are collectively called the system Development Life Cycle (SDLC). The stages in the SLDC are the following.

- Requirements analysis & definition
- System & software design
- Implementation
- Testing
- Maintenance

2.2.1 Requirement Analysis & Definition

All the requirements are gathered at this phase. It is done by site visits, questionnaires, studying, the existing system and interviewing the potential users and the management. The systems services, constraints and goals are established by consultation with the system users. They are then defined in a manner, which is understandable by both users and development staff.

There are several documents used in this stage.

I. Requirement Definition Document

Define services, which are expected from proposed system using natural language. This document must be prepared to easily understand even by no technical person.

II. Software Requirement Specification (SRS) Document

III. Detailed Software Requirement Specification (DSRS) Document

2.2.2 System & Software Design

The system design process partitions the requirements to either hardware or software systems. It establishes overall system architecture. Software design involves representing the software system functions in a form that they may be transformed in to one or more executable programs.

In general, the design process is divided into six different activities.

- Architectural design
- Abstract specification
- Interface design
- Component design
- Algorithm design

There are several strategies for software design. These can be broadly categorized in to two.

I. Functional-oriented design

II. Object-oriented design

Functional-oriented design approaches

- Top-down designing approach
- Bottom-up designing approach
- Jackson structured designing

During this stage, designer employee a variety of tools such as data flow diagrams, entity relationship diagrams, data dictionaries (list of terms and their definitions), and the Gantt chart (a graph depicting design events, personal assignment and time schedules).

Data Flow Diagram (DFD)

When information waves through a software system the data entering the system is modified by a series of transformation. A DFD is a graphical method illustrating the flow and the transformations that are applied as data moves from input to out put.

2.2.3 Coding

The software designed as documents are translated in to programming language.

2.2.4 Software Testing

The program, which was coded, must be tested for possible logical or system errors.

There is a set of fundamental objectives for software testing

- Testing is a process of executing a programme with the intent of finding an error.
- A good test case is one of that has a high probability of finding an as-yet undiscovered error.
- A successful test is one that uncovers as-yet undiscovered errors.

2.2.5 Software Maintenance

The process of changing a system after it has been delivered and is in use is called software maintenance.

All activities carried out for changing a system to maintain its ability to survive is known as maintenance activities. There are different types of maintenance activities as follows;

- Corrective maintenance – fixing reported errors
- Adaptive maintenance – changing the system to some new environment
- Perfective maintenance – implementing new functional or nonfunctional system requirements

2.3 Database Design

2.3.1 Database system

A database is a collection of related data. Data means the known facts that can be recorded and that have implicit meaning. Database and database system have become an essential component at everyday life in modern society.

In traditional database applications most of the information that is stored and accessed is either textual or numeric. In the past few years, advances in technology have been leading to exciting new applications of database systems. Multimedia databases can now store pictures, video clips and sound messages. Geographic information systems (GIS) can store and analyze maps, weather data, and satellite images. Data warehouses and online analytical processing (OLAP) systems are used in many companies to extract and analyze useful information from every large database for decision-making. Real-time and active database technology is used in controlling industrial and manufacturing processes. And database search techniques are being applied to the word wide web to improve the search for information that is needed by users browsing through the Internet.

A database management system (DBMS) is a collection of programs that enables users to create and maintain a database. The DBMS is hence a general-purpose software system that facilitates the process of defining, constructing and manipulating database for various applications. Defining a database involves specifying the data types, structures, and constraints for the data to be stored in the database. Constructing the database is the process of storing the data itself on some storage medium that is controlled by the DBMS. Manipulating a database includes such functions as querying the database to retrieve specific data, updating the database to reflect changes in the miniworld, and generating reports from the data.

The database and DBMS software together is called the database system. Figure 2.4 illustrates the idea.

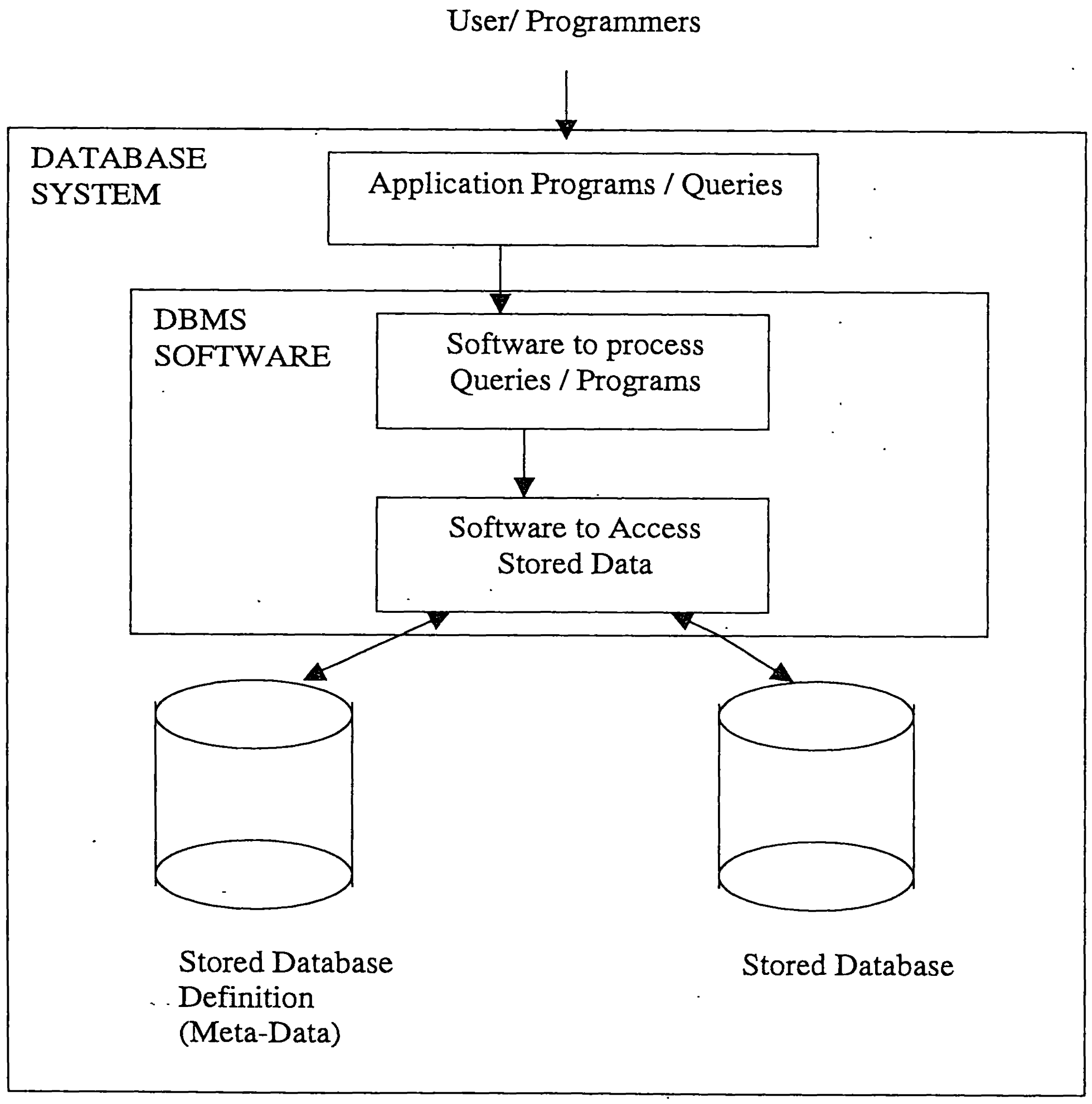

User/ Programmers
DATABASE SYSTEM
Application Programs / Queries
DBMS SOFTWARE
Software to process Queries / Programs
Software to Access Stored Data
Stored Database Definition (Meta-Data)
Stored Database

Figure 2.4 A simplified database system environment
(Source - Elmasri and Navathe, 2000)

2.3.2 Categories of Data Models

A data model is a collection of concepts that can be used to describe the structure of a database.

Data models can be categorized according to the types of concepts they use to describe the database structure.

- Conceptual (High level) data models – provides concepts that are close to the way many user perceive data.
- Physical (low level) data models - provides concepts that describe the details of how data is stored in the computer.
- Representational (implementation) data models - provide a concept that may be understood by end user but that are not too for removed from the way data is organized within the computer. This data models are the data models used most frequently in traditional commercial DBMS, and they include the widely used;
  - Relation database model
  - Network model
  - Hierarchical model

2.3.3 Data modeling using the Entry-Relationship model

Conceptual modeling is an important phase in designing a successful database application. Modeling concepts of the Entry-Relationship (ER) model is a popular high-level conceptual data model. This model and its variations are frequently used for the conceptual design of database application and many database design tools employ its concepts.

ER-Diagram - is used to map the ER-Model. This diagram depicts the ER- Model's three main components; entities, attributes and relationships.

Entity – represents a real word object or concept.

Attribute – represents some property of interest that further describes an entity.

Relationship – represents an interaction among the entities.

Entities are represented by rectangles, while attributes by ovals in ER- modeling.

Tables - The logical view of the relational database is facilitated by the creation of data relationships based on a construct known as a table.

Keys - Keys are attributes, which uniquely define entities.

The ER- model is a tool that commonly used to;

- Translate different views of data among managers, users and programmers to fit into a common framework.
- Define data processing and constraint requirements to help us meet the different views.
- Help implement the database.

2.3.4 Normalization

Normalization is derivation of data as a set of non-redundant, consistent and inter-dependent relations. It is a set of data design standard, which has a process of decomposing unsatisfactory relations into smaller relations.

Advantages of Normalization

Reduction of data redundancy within tables leads to;

- Reduce data storage space
- Reduce inconsistency of data
- Reduce update cost
- Remove many-to-many relationship
- Improve flexibility of the system

Disadvantages of Normalization

Reduction in efficiency of certain data retrieval as relations may be joined during retrieval.

- Increase join
- Increase use of indexes: storage (keys)
- Increase complexity of the system

Normal Forms

Normalization works through a series of stage call normal form. The firs three stages are described as First normal form (INF), Second normal form (2NF), Third normal form (3NF). For most business database design purposes, third normal from is as high as we need to go in the normalization process. (Elmasri and Navathe, 2000)

0NF multi-valued attributes exist

1NF any multi-valued attributes have been removed

A relation is in first normal form if all its attributes are atomic. i.e.

- If there are no repeating groups
- If each attribute is a primitive
- Non decomposable and single valued data items

2NF any partial functional dependencies have been removed

A relation is in second normal form if it is in 1NF and every non-key attribute is dependent on the whole key.

3NF any transitive dependencies have been removed

A relation is in third normal form if it is in 2NF and each non-key attribute is only dependent on the whole key, and not depend on any non-key attribute. i.e. No transitive dependencies.

4NF any multi-valued dependencies have been removed

5NF any remaining anomalies have been removed

2.4 User Interface Designing

The user interface of system is often the yardstick by which that system is judged. Although text-based interfaces will remain in use for many years, users increasingly expect application systems to have some form of graphical interface (GUI).

The advantages of GUI are,

- They are relatively easy to learn and use.
- The user has multiple screens (Windows).for system interaction.
- Fast, full-screen interaction is possible with immediate access to anywhere on the screen.

User Interface Design Principles

- User familiarity – The interface should use terms and concepts which are drawn from the experience of the anticipated class of users.
- Consistency – The interface should be consistent in that comparable operations should be activated in the same way.
- Minimal surprise – User should never be surprised by the behavior of the system.
- Recoverability – The interface should include mechanism to allow users to recover from their errors.
- User guidance – the interface should incorporate some form of context-sensitive user guidance and assistance.

(Sommerville, 2000)

Event –Driven Programming

An **event** is an activity that occurs during a program's execution, such as a mouse click or a keystroke. **Event-driven programming** applies to programming that responds to Windows events. Not only must your programme handle random events, but also if more than one Windows program is running at once, each program needs to analyze and responds to events. In event-driven programming, it allows a user to create Windows applications without using many codes as in any programming language. Programming this way means that when an event occurs, there should be some codes that will make the programme do something in response to the event. Event procedures are block of codes that can be called from the application to trigger an event. This procedure codes might be used to move objects around the form, calculate a value from a formula, or write data to a database. There should be separate event procedures for all events you want to handle and for each control in the window. (Perry, 1998)

Database Access Methods in VB6

Data Access Objects (DAO) model

The DAO model is optimized for accessing Access-Style ISAM databases over local area network connections. Visual Basic 6 ships with two DAO libraries;

- DAO 3.51 Object Library (the most current version)
- DAO 2.5/3.51 Compatibility Library

DAO Jet Database Engine

The idea behind DAO Jet is that you can use one interface to access multiple types of data. Microsoft designed DAO Jet to present a consistent interface to the user regardless of the type of data the user is working with. DAO Jet engine has set of routines and talks to set of translation routines, which convert the request in to a format that the target database can understand. DAO Jet has its own query engine, which can use standard SQL statements to search, order and filter the data.

Remote Data Object (RDO)

RDO is designed for reading and updating data stored in relational database management systems (RDBMS) that are external to Visual Basic and to the Microsoft t Jet data engine. The RDO model is a set of programming objects similar to the DAO model, but there are some important differences, too. One of the most obvious differences is in the size of the object models. The DAO model has over twenty major objects to work with and the RDO model has less than fifteen.

ActiveX Data Objects (ADO)

The Microsoft Data object model can be thought of as a slimmed-down version of the RDO model. But they are not related in some special way, one of the important differences between ADO and RDO is that ADO is built to use the OLEDB interface as the underlying data provider instead of ODBC. With OLEDB "under the covers" ADO is also able to support recordset access to non-SQL data stores, such as email, AS400 and even network directory services.Speed and simplicity is one of the key objectives of ADO. The model is design to create a Recordset object without having to create and navigate numerous other intervening objects along the way.
In fact there are only tree key objects in the model:

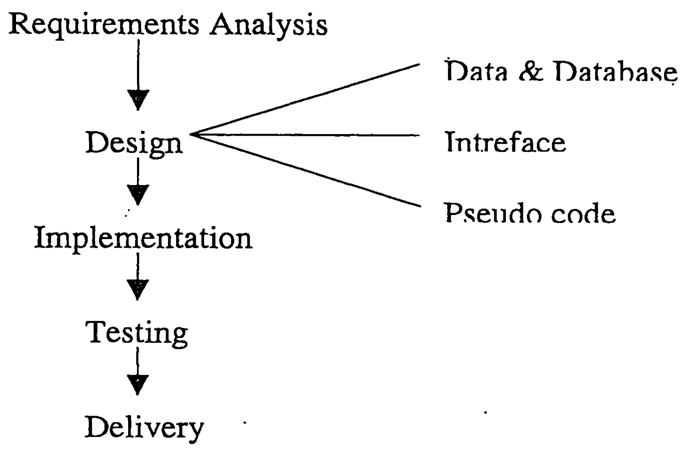
- Connection represents the actual database connection.
- Command is used to execute queries against the data connection.
- Recordset represents the set of records collected from the query issued via the command object. (Smith and Amundsen, 1999)

Chapter 03

3. METHODOLOGY

First of all the nature of the computer system to be built was identified. The type of the system, the functions it should do as well as what are the inputs and outputs were identified.

Then the requirements were analyzed. The requirements were clear, not changed frequently and the proposed system was going to be used for a long time. Therefore the waterfall model was used to development process. The entire development process was carried out as follows;



3.1 Requirements Analysis

All the requirements were gathered by,

- Studying the existing system
- Having discussions with the management
- Interviewing the potential users of the system
- Collecting the documents come in and going out of the system
- Collecting the reports generated by the system

3.1.1 Functional Requirements

1. The System should able to store data of TDR, which are coming in and going out of the system.

1.1 System should automate the TDR data entry, i.e. read the incoming TDR in the discharging process and should write data into the database.

1.2 The user should be provided with facility to enter information on TRD in the process of local loading.

1.3 In Any case of failure of automating the TDR, users could be able to manually enter the data.

1.4 Any Vessel can't be in the TDR with out first appearing in the Vessel Schedule.

2. When updating the Master Files data entry should be checked.

   2.1 System should restrict the users with adding null record sets to the tables.

   2.2 System should restrict the users with adding duplicate record sets to the master files when they already exist.

   2.3 System should restrict the users with deleting null record sets.

   2.4 System should restrict the users with deleting records, which are foreign keys to the tables.

3. For a new vessel the vessel details should be first added to the Vessel details file. So the system should not facilitate to enter any vessel to the Vessel schedule if it's not appear in the Vessel Details.

4. System should transfer the automated TDR details in to standard format called Port Transfer Format (PTF), in order to send the details to SLPA.

   4.1 System should provide a way to view the PTF of the TDR

   4.2 System should provide a way to print the PTF of the TDR

   4.3 In the PTF the relevant billing code should be appear with respect to the operator code.

5. System should generate inward, outward and transshipment reports when the user asks to do so.

   5.1 The users should be provided with an interface to enter relevant data such as vessel, voyage and line code in order to retrieve data.

   5.2 There should be a way to view the report and print them

6. The system should abstract data, calculate them and generate invoices.

6.1 System should auto number the invoices with different numbers for different types of invoices such as PHC and Freight.

6.2 The users should be provided with an interface to enter relevant data such as vessel, voyage and line code in order to view or print the invoices.

6.3 System should generate outstanding reports and also the users should be facilitated with a way of view them and print them.

7. System should generate MIS reports and support inquiry

8. The system should provide security

8.1 The employees are categorized into several employee roles, so the system should check against the User Name, Password and the Employee Role when they are trying to log on to the system.

8.2 The System should provide some defined functions to users, which they can work on according to their employee role.

3.1.2 Non-Functional Requirements

Non-functional requirements are constraints on the services or functions offered by the system such as constraints on time, constraints on the development process, standards, etc. The failure to meet them might result in the system being rejected.

- Usability – User interface must be tailored to the capabilities and background of the system users. Much software is not used to its full potential because, the interface makes it difficult to use.
- Efficiency – Should not make wasteful use of system resources such as memory and processors cycles.
- Reliability – should perform as expected by the user and should not fail more often than it is allowed for in the specification.
- Portability – The target environment independence. i.e. the ease with which a product can be transferred from one environment to another.
- Security – Software should secure against; deliberate corruption by unauthorized users and access to sensitive and confidential data in an unauthorized manner.

3.2 Design

After analyzing, the design was carries out considering the requirements and the system background. Function-oriented design approach was used to design the system.

Data and Database Design

The way of data moving from input to the output was studied. Then the data flow diagrams (DFD) were drawn. There were four DFDs including the Context diagram, up to the third level.

The DFDs are shown in Appendix I

Entities, attributes and the relationships were identified by using the description of the entire system. Then the ER-diagram was drawn to the system. It was converted into tables. Then the tables were decomposed and normalized up to third normal form. Primary keys and foreign keys were also defined. And the relationships among them were established.

There were sixteen tables in the database, named as Logtable, CargoDetails, Cpompanies, CompanyDetails, ContainerSizeType, DangerousCargoDetails, FreightRates, Line Details, PHCDetails, PortCodes, PortwiseAgents, ShipperConsigneeNotify, TDR, Transactions, VesselDetails, VesselSchedule

Interface Design

A user friendly, graphical user interface was required to the system. The Visual Basic development environment was proposed by the management for this purpose. So the interface design was carried out considering the user interface design principles and using Visual Basic 6.0 Enterprise Edition.

Forms were designed as the gateway to access the system. There were thirty-eight forms, which were used to input data and retrieve them from the database.

Pseudo code Design

Pseudo codes were designed to each function of the system. Following is an example code of adding VesselDetails .

```
Private sub CommandButton_click()
If "code" equals Null then
  Message box appear
  Set focus to "code"
  Exit sub
End if
If  record count equals 0
  Write the record to the database
  Map the primary key field to the relevant input
Else
   If code already exist Message box appear with asking "do you want to
  update?"
If  No then
    Exit Sub
 End If
End If
  Map database fields to the relevant input data
After adding clear the input boxes
 Set focus to "code"
End sub
```

3.3 Implementation

SQL Server 2000 was used to implement the database.
Appendix II illustrates some of the tables and their fields.

User interfaces were created in Standard EXE format of Visual Basic 6.0.
Some of the system forms are shown in Appendix III.

And the VB codes were generated to trigger the events. In order to access the database and manipulate data, ActiveX Data Objects (ADO) model was used.

Making the Connection String

The ADO model uses the ConnectionString property to indicate the OLE DB provider to use to connect to the data store, along with all the details needed to complete the data connection.

Connection String, which was used to connect the database as follows,

Provider=SQLOLEDB.1; Password=abc; Persist Security Info=True; User ID=sa; Initial Catalog=SeaS; Data Source=MAX-X\SQL

Appendix IV illustrates some of the codes generated to update the database.

3.4 Testing and Delivery

Testing was done with the intention of finding errors or bugs in the system. It was done as unit test, system test and acceptance test. Unit test was carried out when the codes were written. Then the system test was done to ensure the integrity of each module in the system to act as a single unit. Finally the overall acceptability and compatibility of the new system was tested with the participation of management of the Green Lanka Shipping Ltd.

After the acceptance of the new system, some employees were trained for the system.

Chapter 04

4. RESULTS AND DISCUSSION

4.1 Results

The outcome of the system was a user-friendly information system to handle information related to Feeder Service of the Sea Services (PVT) Ltd. The system met all the user requirements except the MIS reports. And also the drawbacks of existing system were overcome.

4.2 Discussion

When developing the system more effort was done on automating the TDR data entry. It was not an easy task since an Excel work sheet has to be read and written into the database.

Use of SQL Server to implement the database was more important to implement the security functions of the system.

The functions of generating MIS reports and Inquiry were not implemented due to time constraints.

But the overall system meets almost all the requirements and it is up to the expected standards hence overall acceptability for the entire system is in satisfactory level.

Chapter 05

5. CONCLUSION AND RECOMMENDATION

According to the stakeholders' point view the objective of developing a Windows base user-friendly system has been achieved. This was achieved by following the proper software development methodologies.

Further Enhancements

- Introducing online access features to the system will make it more advances. This facility provides access to the system more easily from everywhere.
- Introducing functions to support inquiries and generate MIS reports. This facility provides report generating and inquiring for the top-level management.
- Use of a report generating tools such as Seagate Crystal Reports will enhanced the quality of reports.

References

Elmasri, R. and Navathe, S.B. (2000) Fundamentals of Database Systems. 3rd Ed, Addison Wesley an imprint of Pearson Education Inc. 955p.

Perry, G. (1998) Sams Teach Yourself Visual Basics 6 in 21 Days.1st Ed, Techmedia, 851p.

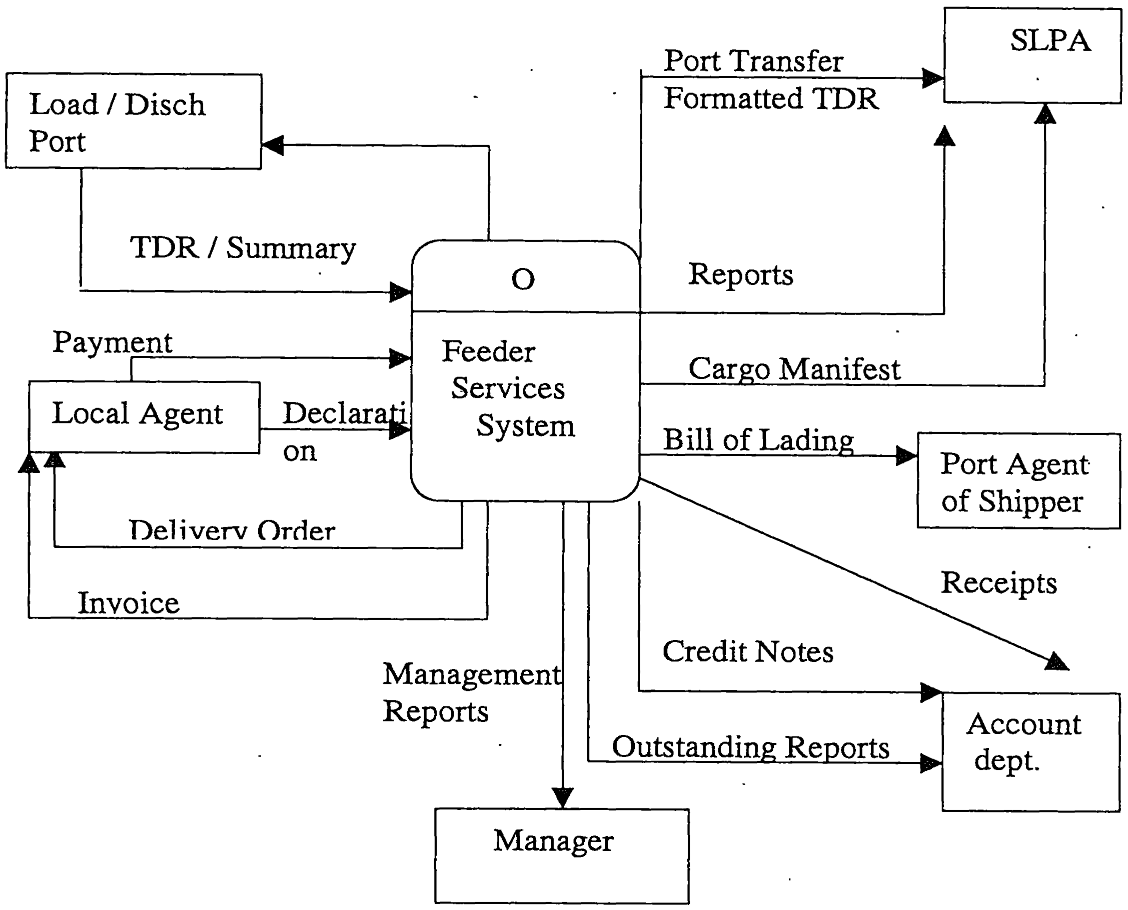
Smith, C. and Amundsen, M (1999) Sams Teach Yourself Databaes Proframming with Visual Basics 6 in 21 Days.1st Ed, Techmedia, 882p,

Sommerville, I. (2000) Software Engineering. 5 th Ed, Addison Wesley Logman (Singapore) Pte.Ltd , 742p.

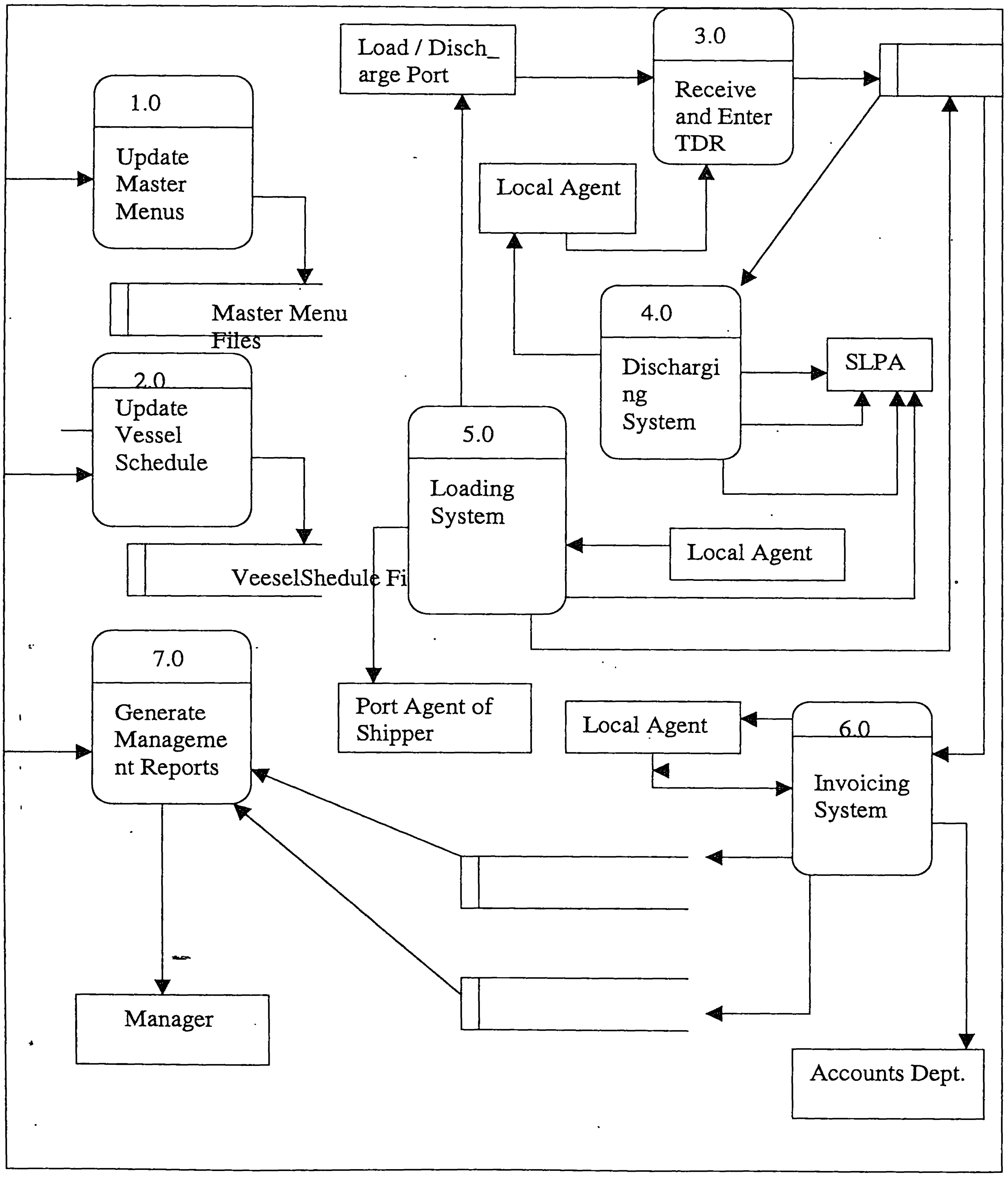
Appendix I

Data Flow Diagram

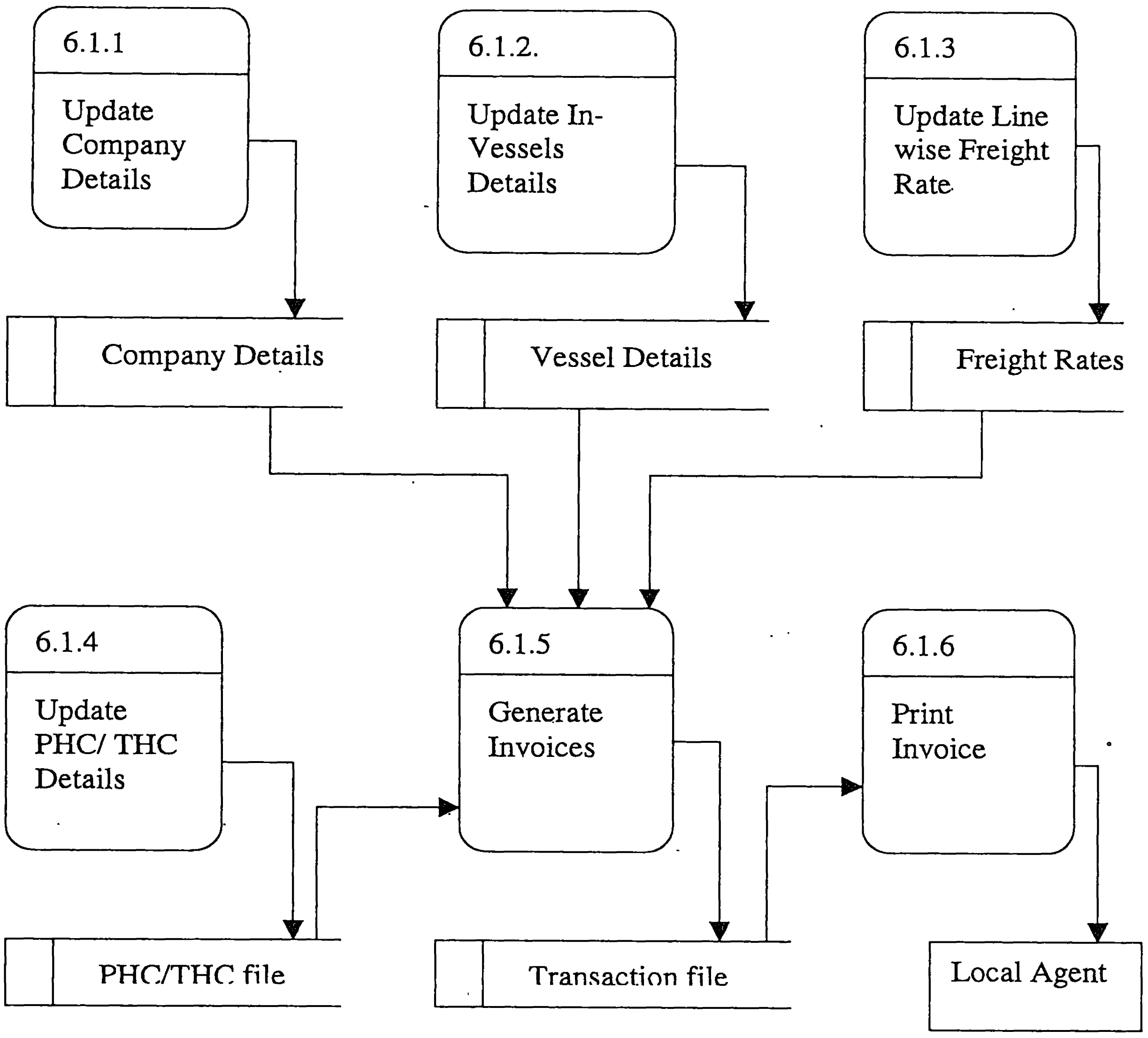
Context Diagram



Level 1 DFD of Feeder Services System



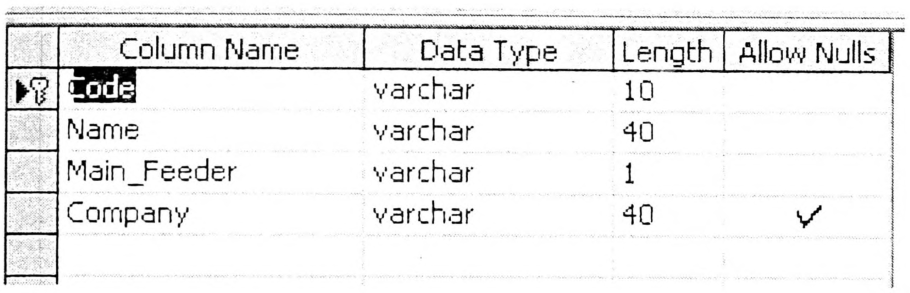
Level 3 diagram showing the decomposition of process 6.1 from the Level 1 diagram



Appendix II

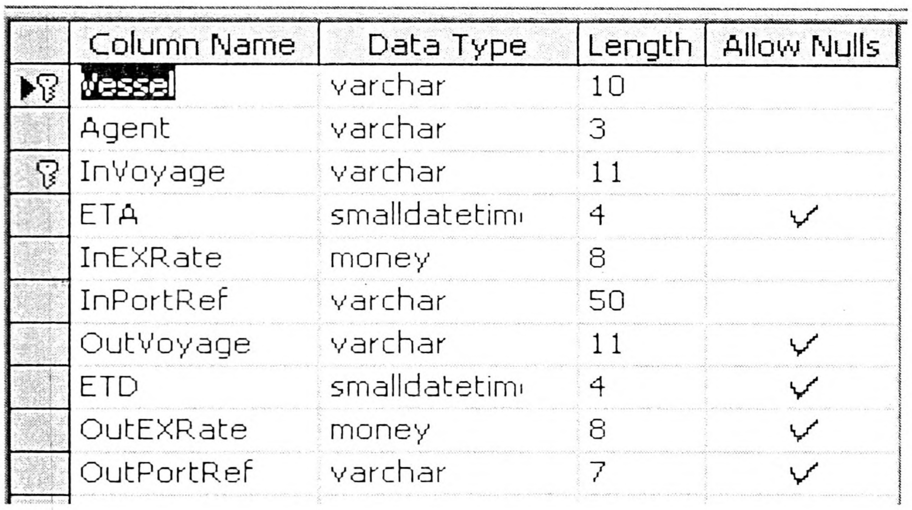
Table Structures

Table 1.Vessel Details



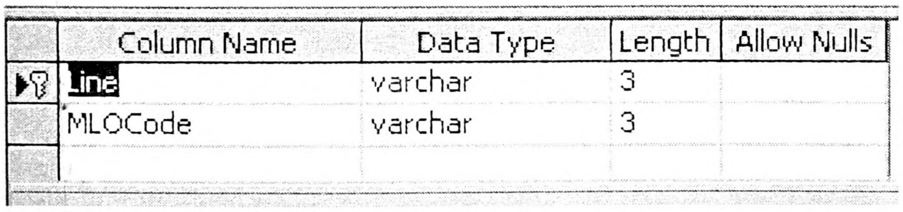
	Column Name	Data Type	Length	Allow Nulls
▶🔑	Code	varchar	10	
	Name	varchar	40	
	Main_Feeder	varchar	1	
	Company	varchar	40	✓

Table 2.Vessel Schedule



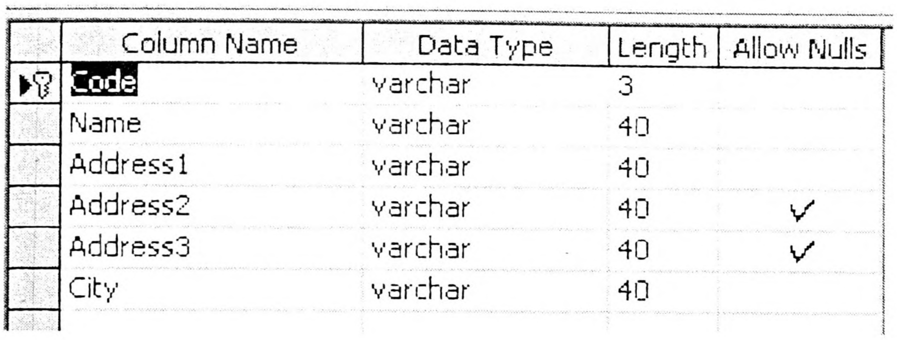
	Column Name	Data Type	Length	Allow Nulls
▶🔑	Vessel	varchar	10	
	Agent	varchar	3	
🔑	InVoyage	varchar	11	
	ETA	smalldatetim	4	✓
	InEXRate	money	8	
	InPortRef	varchar	50	
	OutVoyage	varchar	11	✓
	ETD	smalldatetim	4	✓
	OutEXRate	money	8	✓
	OutPortRef	varchar	7	✓

Table 3.Line Details



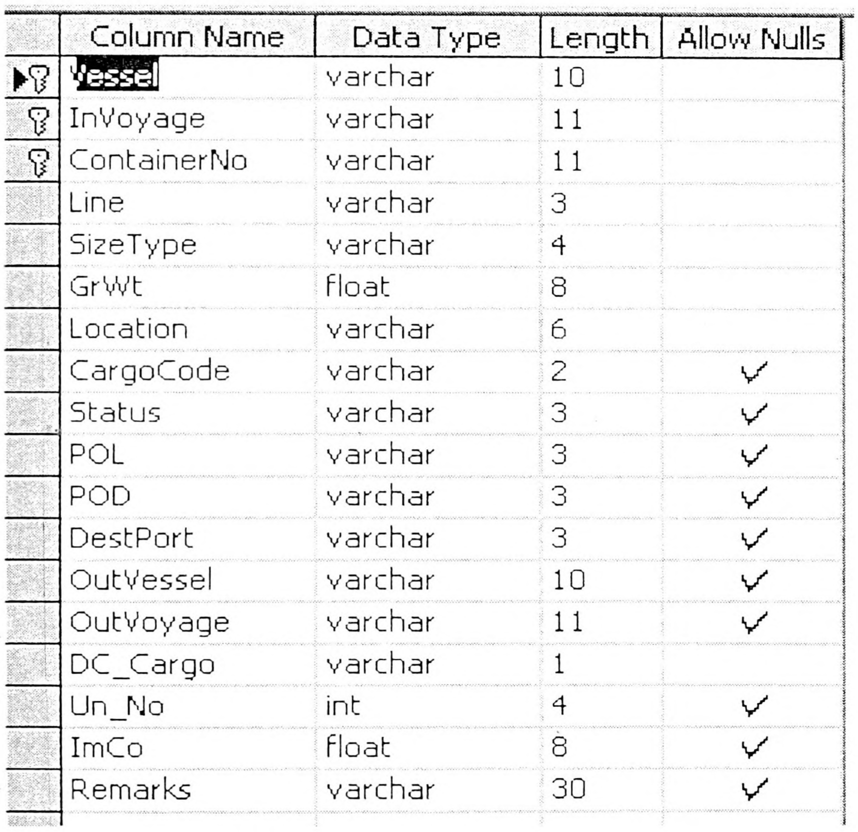
	Column Name	Data Type	Length	Allow Nulls
▶🔑	Line	varchar	3	
	MLOCode	varchar	3	

Table 4. Company Details



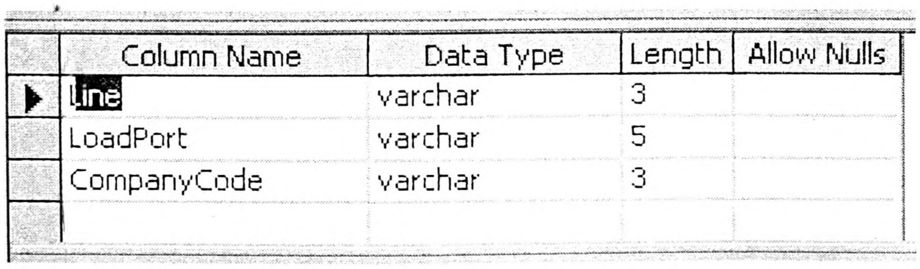
	Column Name	Data Type	Length	Allow Nulls
🔑	Code	varchar	3	
	Name	varchar	40	
	Address1	varchar	40	
	Address2	varchar	40	✓
	Address3	varchar	40	✓
	City	varchar	40	

Table5. TDR



	Column Name	Data Type	Length	Allow Nulls
🔑	Vessel	varchar	10	
🔑	InVoyage	varchar	11	
🔑	ContainerNo	varchar	11	
	Line	varchar	3	
	SizeType	varchar	4	
	GrWt	float	8	
	Location	varchar	6	
	CargoCode	varchar	2	✓
	Status	varchar	3	✓
	POL	varchar	3	✓
	POD	varchar	3	✓
	DestPort	varchar	3	✓
	OutVessel	varchar	10	✓
	OutVoyage	varchar	11	✓
	DC_Cargo	varchar	1	
	Un_No	int	4	✓
	ImCo	float	8	✓
	Remarks	varchar	30	✓

Table 6. Port wise Agents Details

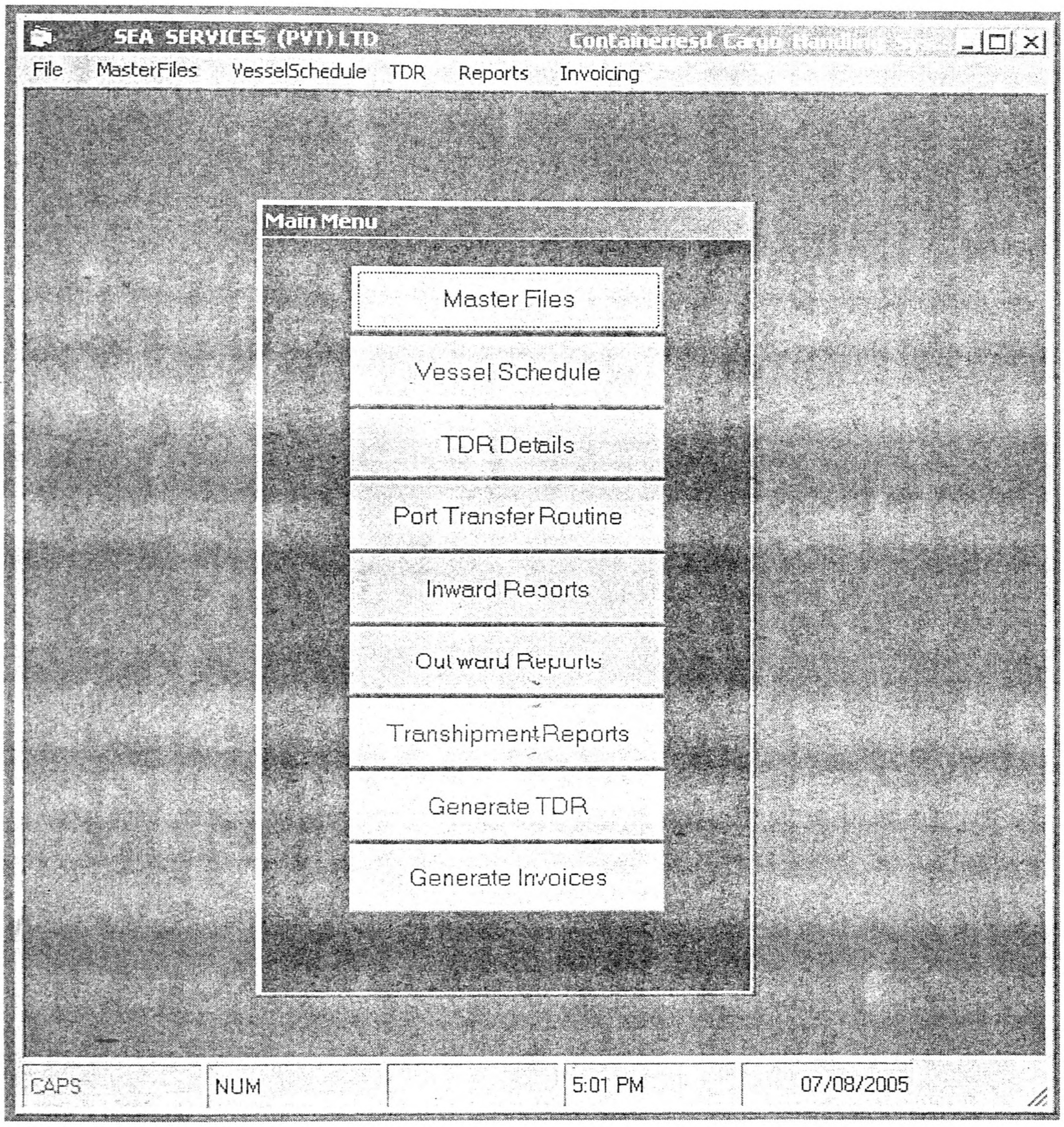


Column Name	Data Type	Length	Allow Nulls
Line	varchar	3	
LoadPort	varchar	5	
CompanyCode	varchar	3	

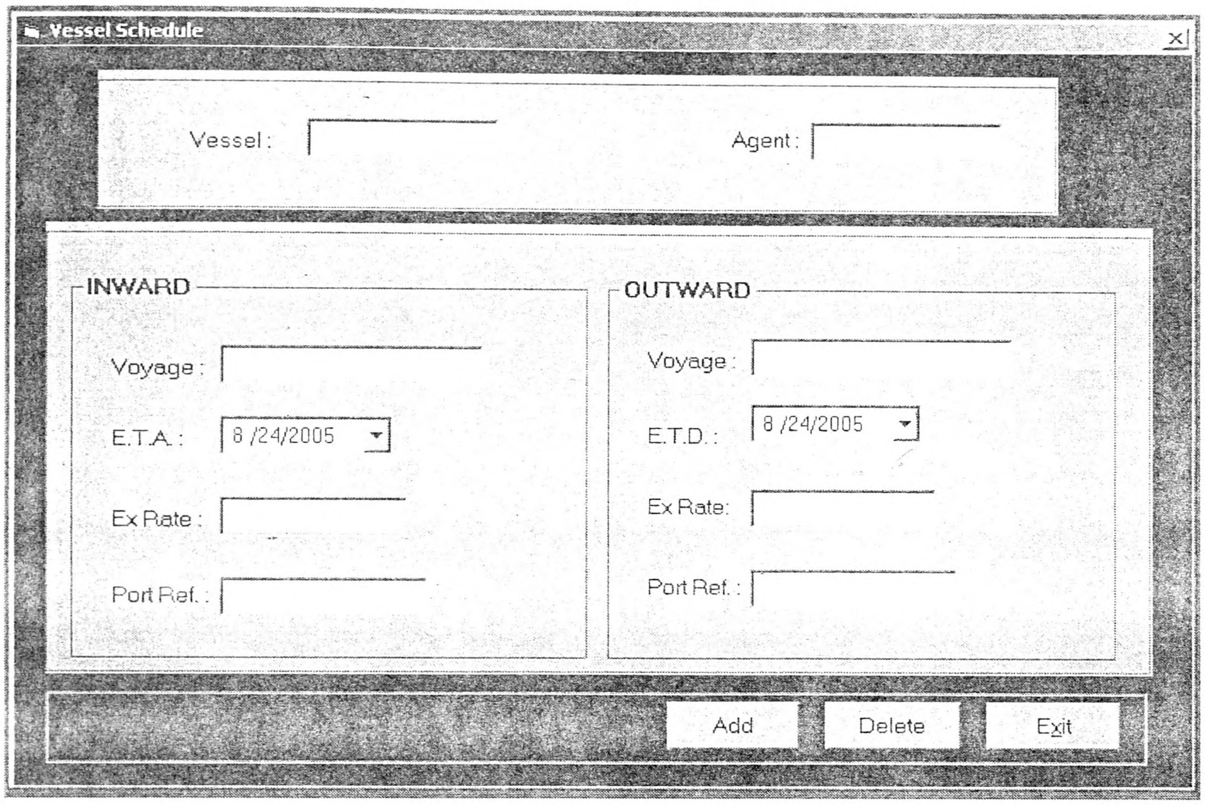
Appendix III

User Interfaces

Form 1. Main Menu



Form 2. Vessel Schedule



Vessel Schedule

Vessel :

Agent :

INWARD

Voyage :

E.T.A. : 8 /24/2005

Ex Rate :

Port Ref. :

OUTWARD

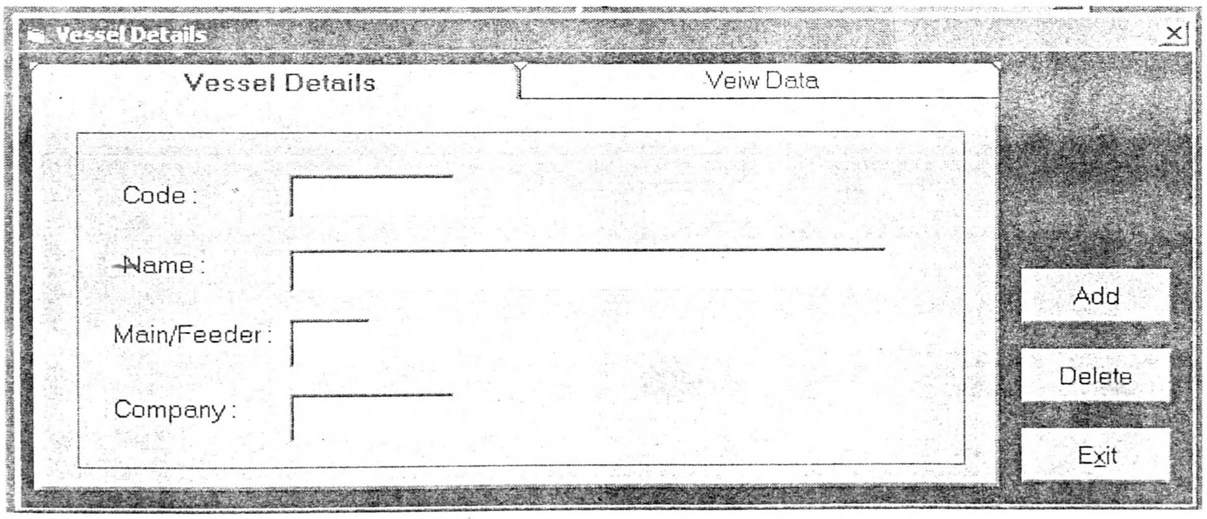
Voyage :

E.T.D. : 8 /24/2005

Ex Rate:

Port Ref. :

Form 3. Vessel Details



Vessel Details

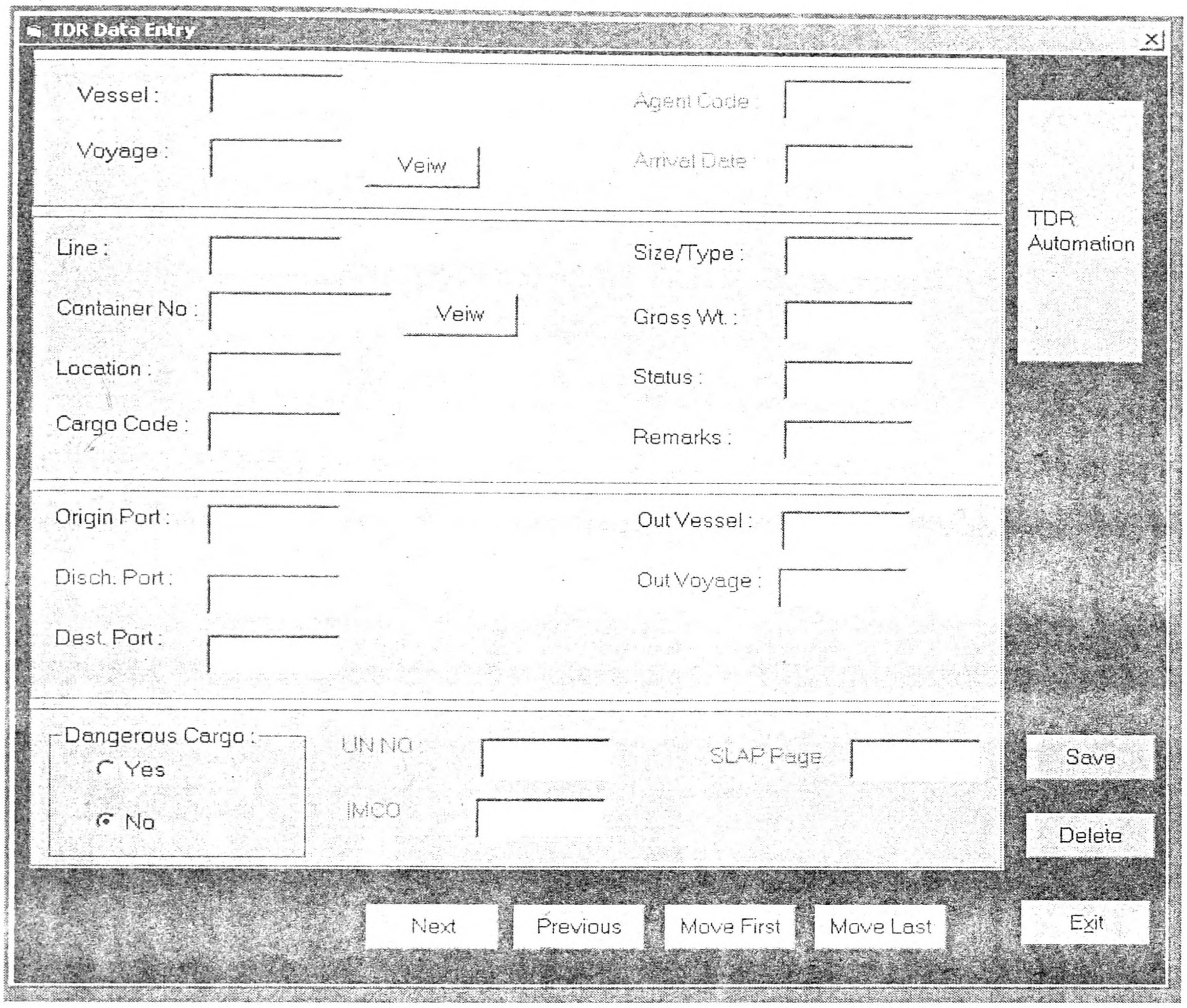
Code :

Name :

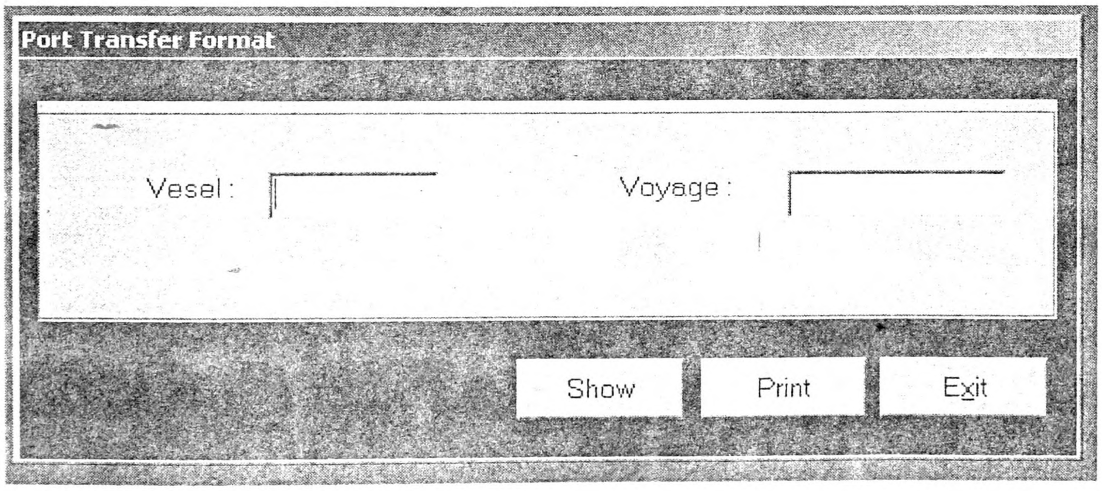
Main/Feeder :

Company :

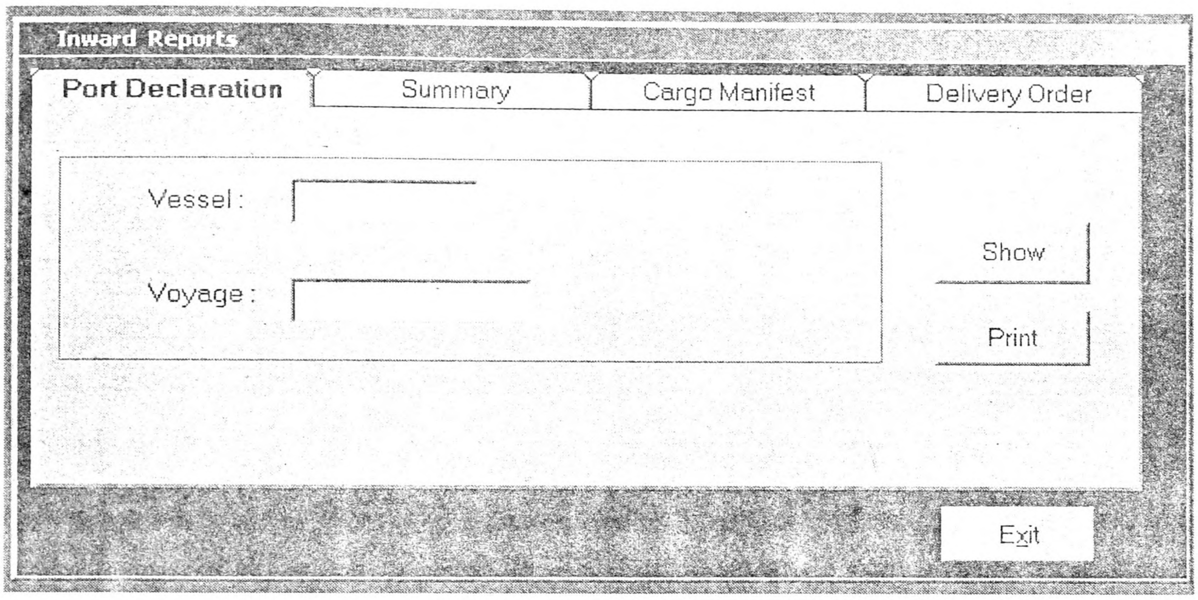
Form 4. TDR Data entry

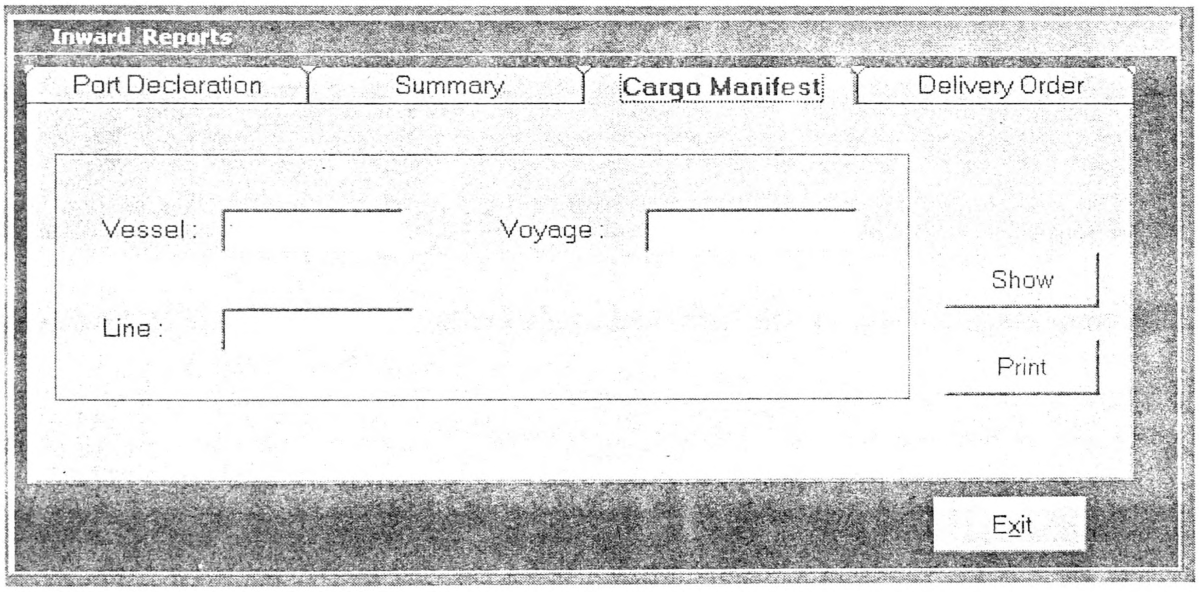


Form 5. Port Transfer Routine



Form 6. Generate Inward Reports





Appendix IV

Codes of the system

e.g. VesselDetails table

To Add Records

```
Private Sub cmdAdd_Click ()
ssq = "SELECT  *  From VesselDetails"
ssq = ssq & " Where Code='" & txtCode & "'"
adoVes.RecordSource = ssq
adoVes.Refresh
Dim intKey As Integer
If  txtCode.Text = ""  Then
   intKey = MsgBox("Incomplete data to Add", vbOKOnly + vbExclamation)
Else
  If  adoVes.Recordset.RecordCount > 0   Then
     adoVes.Recordset.MoveFirst
     While Not adoVes.Recordset.EOF
          If  UCase(txtCode.Text) = adoVes.Recordset!Code  Then
             intKey = MsgBox("Code already Exists !", vbOKOnly + vbExclamation)
          End If
          adoVes.Recordset.MoveNext
      Wend
      adoVes.Recordset.MoveLast
  Else
      adoVes.Recordset.AddNew
      adoVes.Recordset!Code = UCase(txtCode)
      adoVes.Recordset!Name = UCase(txtName)
      adoVes.Recordset!Main_Feeder = UCase(txtMF)
      adoVes.Recordset!Company = UCase(txtComp)
       adoVes.Recordset.Update
  End If
End If
  Call prcClear
  txtCode.SetFocus
End Sub
```

To Delete Records

```
Private Sub cmdDel_Click()
Dim ctr As Integer
ctr = 0
Dim vntVal As Variant

If  txtCode.Text = ""  Then
   MsgBox "No Data to Dalete", vbOKOnly + vbExclamation
Else
   If  MsgBox("Are You sure that you want to delete the current Record?", _
     vbYesNo + vbQuestion, "DELETE") = vbYes  Then
     Err.Clear
     adoSch.Recordset.MoveFirst
     While Not adoSch.Recordset.EOF
          If  UCase(txtCode.Text) = adoSch.Recordset!Vessel  Then
             vntVal = 1
             ctr = 0
          End If
          adoSch.Recordset.MoveNext
     Wend
      If  vntVal = 1 Then
          intKey  =  MsgBox("You  can't  delete  that  record,since  it  appears  on
          VesselSchedule" ,  vbOKOnly + vbInformation)
      Else
          adoVes.Recordset.Delete adAffectCurrent


           adoVes.Recordset.MovePrevious
     End If
      End If
End If
txtCode.SetFocus
```

```
 Call prcClear
   End Sub
```

To Exit from the form

```
Private Sub cmdExit_Click()
  Unload Me
End Sub
```

Common procedure to every form

```
Public Sub prcClear()
For Each Control In Me
   If TypeOf Control Is TextBox Then
       Control.Text = ""
   End If
Next
End Sub
```

KeyPress event of the VesselDetails

```
Private Sub txtCode_KeyPress(KeyAscii As Integer)
If  KeyAscii = 13  Then
    ssq = "SELECT  *  From  VesselDetails"
    ssq = ssq &  " Where Code='"  &  txtCode  &  "'"

    adoVes.RecordSource = ssq
    adoVes.Refresh
   If  adoVes.Recordset.RecordCount > 0  Then
       txtCode = adoVes.Recordset!Code
       txtName = adoVes.Recordset!Name
       txtMF = adoVes.Recordset!Main_Feeder
       txtComp = adoVes.Recordset!Company
   End If
End If
End Sub
```