UNIVERSITY OF NAIROBI

COLLEGE OF BIOLOGICAL AND PHYSICAL SCIENCES

SCHOOL OF COMPUTING AND INFORMATICS

CHIROMO CAMPUS

‘D-Mash’ - A CONCURRENT DISTRIBUTED BOOKING SYSTEM

A RESEARCH PROJECT REPORT SUBMITTED TO THE SCHOOL OF COMPUTING AND INFORMATICS IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF MASTER OF SCIENCE IN DISTRIBUTED COMPUTING TECHNOLOGY OF THE UNIVERSITY OF NAIROBI

NAME: NGUMBAU SHADRACK MWANZIA

REGISTRATION NUMBER: P53/65245/2013

SUPERVISOR: DR.ANDREW MWAURA KAHONGE

DEGREE PROGRAMME

MASTER OF SCIENCE (DISTRIBUTED COMPUTING TECHNOLOGY)

DECLARATION

This research project is my original work and has not been presented for a degree in any other University.

Student: Shadrack Mwanzia Ngumbau (P53/65245/2013)

Date: ……..

Signature: ……..

This research project has been submitted for examination with my approval as University supervisor.

Supervisor: Dr.Andrew Mwaura Kahonge

Date: ……..

Signature: ……..

ABSTRACT

Since time immemorial, man has always looked for a better way to co-exist with nature in a diverse universe. In pursuit of this, challenges such as transport problems in remote areas, latency in communication channels amongst others have been met. Today's period reflects an era in which the universe is a digitally connected global village. Researchers and scientists have devised ways of solving communication and movement problems to achieve co-existence. Communication has now been made possible through technologies such as teleconferencing without moving or meeting in a central place but as distributed entities all over the world.

Despite this happening, new technologies come with challenges hence communication in a distributed and concurrent context is a problem. The aims of this project was to investigate the impact of automated distribution and concurrency on Service Quality and develop a practical solution to a real-world concurrency problem in a distributed bus seat booking environment which also acted as an aid in our investigation.

This solution was a concurrent distributed booking prototype that was developed for a local Bus Company. It employed the action research design where data was collected and analysed, from a sample size of 1 bus company amongst a population of approximately 18 reputable bus companies and 5 respondents amongst a sample population of approximately 8 respondents, using the simplified random sampling technique.

From the analysis of the results, it was found out that the probability of double-booking a seat in a distributed environment was reduced to 0 hence achieving concurrency which improved the quality of service rendered using the system that was developed using distributed computing and concurrency techniques.

Keywords: Distributed Systems; Action Research; Web Services; Concurrency Control; Middleware; Synchronisation

DEDICATION

I dedicate this report to my parents Mr.Armstrong Ngumbau Mwanzia and Mrs.Susan Ngumbau who have motivated and supported me holistically in my academic undertakings.

ACKNOWLEDGEMENTS

I would like to acknowledge my supervisor (Dr.Andrew Mwaura Kahonge) and panellists (Prof.Timothy Waema, Dr.Elisha Abade, Mr.Eric Ayienga, Prof.Elijah Omwenga) who have given me an outline of what I am supposed to do and who have guided me during the project. They were there to give advice, comments, suggestions, and corrections where necessary in order to make the project a success. I would also like to acknowledge the ICT Technicians and the University of Nairobi's fraternity for offering good services to me.

In addition, I'd like to thank my Pastor, Samuel Mwanzia who also supported me in prayer and in provision of a laptop that was required at the research site during the study. I also thank everyone who was there to help me in this project. Above all, I thank God for everything.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

SOAP – Simple Object Access Protocol

WSDL – Web Service Description Language

XML – Extended Mark-up Language

UDDI – Universal Description Discovery and Integration

API – Application Programming Interface

CORBA – Common Object Request Broker Architecture

ORB – Object Request Broker

IDL – Interface Definition Language

EJB – Enterprise Java Beans

RMI – Remote Method Invocation

CLR – Common Language Runtime

CTS – Common Type System

CLS – Common Language Specification

Chapter One

1.0 INTRODUCTION

1.1 Background

In this day and age, technology has drastically brought about a revolution in the way activities are executed. This is clearly depicted by the manner in which entities are able to communicate and collaborate with fewer communication hitches as compared to the past regardless of the location of the entity. This has been made possible by the on-going research and implementation of Distributed Systems.

However, as stated by Thomas and Mukesh (1994), issues such as fault tolerance, crash recovery, high-performance, real-time computing, and synchronisation remain to be a challenge to **distributed** system designers and researchers. To achieve synchronisation, solutions have been sought for such as incorporating concurrency within distributed systems. It is through implementation of distributed algorithms that mutual exclusion, which tends towards solving the synchronisation issue, is attained.

This project seeks to solve a concurrency and distribution problem in a bus seat booking environment using a concurrent distributed booking prototype developed as an aid in investigating the impact of automated distribution and concurrency on Service Quality. It also entails researching into the challenges faced by the current technologies to achieve distribution and trying to solve them by implementing and testing a system based on a proposed framework to counteract them. These are challenges such as presence of bottlenecks in centralised systems, data format incompatibilities in middleware technologies amongst others.

1.2 Problem Statement

In pursuit of development and making progress as a country, the people of Kenya have learned to adopt and embrace technology over the years since colonisation. Improvements have been made in many sectors technology and transport being some of the sectors that have experienced great improvements. In these sectors, distributed systems and telephony are used to help achieve commercial communication.

It is as a result of this that mobile phones exist that are used today by booking officers to communicate with each other so as to ensure that no bus seat is booked twice in different offices. This is done in their pursuit of concurrently booking seats while in different locations so as to achieve synchronisation. This has worked to some extent but with difficulties too leading to instances of a seat being booked twice unexpectedly.

To prevent such instances, this research project seeks to solve a concurrency and distribution problem in a bus seat booking environment using a concurrent distributed booking prototype developed for officers as an aid in investigating the impact of automated distribution and concurrency on Service Quality.

It also seeks to help an organisation solve its problems and improve on its key attributes such as productivity, the quality of their products and/or services, and working conditions (Kock, 2004). Automated concurrency and distribution techniques are introduced and their impact on service quality tested to help solve the problem in addition to supporting a valid research conclusion.

1.3 Research Objectives

The main objective is to determine the impact of automated distribution and concurrency on Service Quality with an aid of a distributed concurrent booking system.

The specific objectives are;

i. To explore the technologies and algorithms used to implement concurrent distributed systems.
ii. To determine the efficiency of the implemented system and perception of users towards it.
iii. To determine the kind of effect on the booking process as a result of introducing a concurrent distributed system.

1.4 Research Questions

i. Which technologies and algorithms are used to embed concurrency in distributed systems, what are their drawbacks and advantages of one over the other?
ii. How efficient is the system and what's the perception of the users towards it?
iii. Is there a positive or negative effect on the booking process and service offered as a result of the system's use?

1.5 Justification

The research brings with it a solution that is developed using well known techniques and tested leading to a published successful method employed to develop a concurrent distributed booking system. An improvement in research and development through the use of distributed computing means saving time and costs spent on researching, developing, and testing of a concurrent distributed system from scratch.

An identified and published method and framework are made available to adopt or improve on. In addition, this research enables reflection to be made on the action research method as one of the methods used in problem solving. Besides that, contemplation is made on the concurrency problem in a distributed environment with a view of coming up with and testing applicability of a framework that can be used by researchers and practitioners as a future basis of solving a related problem.

This research presents a solution for the Bus companies that especially provide long distance transport services. The project presents an opportunity for application of the knowledge gained and attainment of more skills with an aim of satisfying the users. It enables the appreciation of the concepts and theory behind distributed software development and integration in addition to exhibiting the challenges encountered in the development of such a system. It also provides a method employed to develop a distributed system for researchers to critique, reject, or improve with an aim of providing the best solutions to the society.

Chapter Two

2.0 LITERATURE REVIEW

2.1 Distributed Systems

"A Distributed System is a collection of independent computers that appears to its users as a single coherent system" (Tanenbaum and Steen, 2007 p.2) As detailed further in their book, these systems consist of autonomous components which are perceived to be a single image system. In addition, they are heterogeneous and separate but interconnected together to ensure coordination and cooperation to achieve a particular task.

Such systems are transparent to the users a characteristic that enables modifications such as replacement, repair, removal, and upgrading of components e.g. servers without the users being aware of any changes made on the system (Tanenbaum and Steen, 2007). However, operation and execution of these systems is not always successful because of difficulties such as communication, synchronisation, coordination, advanced heterogeneity problems, etc. whose solutions have been sought for through continuous research.

The introduction of these systems enabled and has continuously narrowed the gap between human beings in today's interconnected digital village. Examples of such are the search engines such as Google and Bing, the internet, Grid Systems e.g. TeraGrid, Cloud computing systems etc. 'D-Mash' is an example that seeks to make a contribution to this by enabling individuals work together remotely without conflicting each other.

As Tanenbaum and Steen (2007) explain, distributed systems exhibit openness, scalability, and fault tolerance amongst other characteristics. They further detail that such systems can adopt layered, data-centred, object-based, and event-based architectures. In addition they can be implemented using centralised, multi-tiered, decentralised, and hybrid system architectures (Tanenbaum and Steen, 2007). The system's (D-Mash's) implementation uses an object-based architecture and a decentralised distributed database platform.

2.1.1 Distributed Databases

One of the important components of existing distributed systems is a distributed database which is "a single logical database that is spread physically across computers in multiple locations that are connected by data communication links" (Kaur and Kaur, 2013, p 1). Moreover, it's a kind of virtual database whose distinct sub-components/fragments exist in real databases located in different computers/servers (Kaur and Kaur, 2013).

The essence of a distributed database is to improve performance, increase reliability and availability, reduce communication costs, provide for distribution and autonomy of business units as well as provide faster response (Kaur and Kaur, 2013). D-Mash consists of a distributed database to support distribution of several booking offices and call for reduction of unnecessary communication costs.

The database is usually implemented based on a distributed database design which considers data fragmentation and sometimes replication (Kaur and Kaur, 2013). Kaur and Kaur (2013) add that fragmentation can be sub-divided into horizontal, vertical, and hybrid fragmentations.

2.1.2 Transaction Management and Concurrency Control

As explained by Kaur and Kaur (2013), transaction management is also another important element of distributed databases which seeks to achieve atomicity, durability, consistency, and isolation. Another issue that goes hand in hand with transaction management is concurrency control which "is the activity of coordinating concurrent accesses to a database in a multi-user database management system" (Kaur and Kaur, 2013, p 1444).

According to their explanation, the main difficulty is encountered when trying to make sure that database updates made by one individual do not collide with other retrievals and updates. This is because there would be integrity and consistency flaws if this is not met.

2.2 Concurrency

To achieve concurrency, several algorithms are implemented e.g. distributed two-phase locking, timestamp ordering, distributed optimistic, and wound-wait algorithms. According to Jitendra and Gupta (2012), the flat transaction model has proved to be a good model for banking and airline reservation systems over the years.

However a new model needs to be established for complex applications and distributed systems (Jitendra and Gupta, 2012). This is the nested transaction model which invokes atomic transactions and operations. It's a model that involves the execution of nested transactions. Nested transactions provide safe concurrency within transactions and allow for parallelism (Jitendra and Gupta, 2012).

A distributed transaction includes one or more statements that refer to or modify data on two or more distinct sites of a distributed database (Jitendra and Gupta, 2012). This could be a transaction that updates a value in one site's database then use that to update another value on a different site. Jitendra and Gupta (2012) explain transaction management as a process of trying to maintain a consistent state even when concurrent accesses and failures occur. This seeks to achieve consistency in databases so as to provide accurate and reliable information.

Jitendra and Gupta (2012) further detail that distributed transactions could be flat or nested transactions where a nested transaction involves a series of operations running within another operation of a flat transaction. Ensuring that concurrency is maintained in distributed databases is a challenge and Jitendra and Gupta (2012) propose locking and timestamping as techniques to be used in solving this. 'D-Mash' seeks to implement locking as one of the techniques to achieve concurrency.

According to Gupta et al. (2011), some of the different aspects considered to achieve concurrency are distributed commit, failure of individual sites, multiple copies of data etc. 'D-Mash' seeks to achieve a distributed commit by ensuring that a commit is done on the proper database and no commit is done on a database whose server is not accessible or has failed. It is through that concurrency that leads to database records' consistency is met.

2.2.1 Wound Waiting (WW)

Quasim (2013) explains that a younger transaction is wounded or restarted in case it tries to prevent an older time stamped transaction unless it is in the second phase of its commit protocol. This is done to prevent deadlocks. It also ensures that the transaction initiated earliest is given a priority as opposed to waiting in the phase locking algorithm.

2.2.2 Timestamp Ordering

This technique is based on the principle that an operation would only be allowed to execute if its timestamp is newer than other transactions' timestamps (Quasim, 2013). A problem would arise if wrong time stamping is done from a transaction's end such that one transaction shares a similar timestamp with another. This means that the two transactions will conflict and may lead to a deadlock. However, this can be solved by ensuring that the nodes from which a transaction originates have their clock systems synchronised to the global real time clock system.

2.2.3 Distributed Certification

In this, transactions can read and write any time as the updates to a database are stored on a local workspace to wait for the commit time (Quasim, 2013). This allows for execution of an operation even when the final atomic update to a database cannot be executed at a particular point in time due to unavailability of a lock that can be acquired. The stored updates are then executed during their commit time according to their read and write timestamps in the workspace.

2.2.4 Distributed two phase locking (2PL)

As introduced above, several algorithms are used to achieve concurrency one of which is two-phase locking. Quasim(2013) states that a transaction may only be allowed to access a piece of data once a lock has been obtained. In distributed databases locking may occur at read or write levels and this is termed as multi-granularity locking in addition to an existing global level (Quasim, 2013). D-Mash implementation seeks to use locking as opposed to wound-waiting, timestamp ordering, and distributed certification because the system provides locks to any

process regardless of the time the transaction was initiated. In this case, there is no younger or older transaction.

2.3 Web Services

According to Tanenbaum and Steen (2007) the World Wide Web is no longer a document-based distributed system but a huge distributed system through which documents as well as services are used, composed, and offered to online users. This is due to the introduction of web services which refers to traditional services offered online via the internet. A web service adheres to standards enabling it to be discovered and used by clients conforming to the same standards (Tanenbaum and Steen, 2007).

Padiya and Mumbaikar (2013) explain that the main goal of web services is to enable exchange of information amongst distributed applications. Web services are highly platform independent, loosely coupled, and the users need not have prior knowledge of them before consuming them (Padiya and Mumbaikar, 2013). This means that they support exchange of information amongst clients and servers of different hardware and software features.

As Gashti (2012) explains, web services are made based on XML (Extended Mark-up Language) which is based on SOAP (Simple Object Access Protocol) to enable them connect and communicate with heterogeneous programs launched in different computers. They are discoverable to external users once they have been registered in a directory that stores service descriptions (Tanenbaum and Steen, 2007). They are able to explain what they do through the WSDL (Web Service Description language) which is an XML-based standard.

Gashti (2012) details that UDDI (Universal Description Discovery and Integration) is a registry standard for web service providers, to publish their web services in form of service descriptions, and web service consumers to search for published web services. UDDI acts as a commercial meeting point for providers and consumers just as a marketplace or online shopping portal would act. Web services assume that services provided by different companies or providers are loosely-coupled since in general every company as a unit defines, develops, and manages its services in its own unique way (Alonso et al., 2004).

With that maintained, they aim to connect such providers together regardless of their diversity. A web service is hence a software application with a stable API (application programming interface) that's made available to consumers (Alonso et al., 2004). 'D-Mash' seeks to implement web services based on SOAP as one of the technologies to help in connecting different servers. Although Padiya and Mumbaikar (2013) prove otherwise, 'D-Mash' aims to implement web services with no assumption of high latency and network traffic for it is at the initial not scaled phase hence a small element of tight-coupling exists.

2.4 Middleware

According to Tanenbaum and Steen (2007) a distributed system is often organised by means of a layer of software known as middleware between applications/clients and servers/operating systems. Several types of middleware exist such as remote procedure call, reflective middleware, adaptive middleware, object-based middleware, and message-oriented middleware.

Ghaleb and Rheda (2012) explain that middleware is an important element for distributed and networked applications in that middleware solutions provide support for functional and non-functional requirements of distributed applications. "Distribution middleware defines higher level distributed programming models whose reusable APIs and mechanisms automate and extend the native OS network programming capabilities encapsulated by host infrastructure middleware" (Schmidt, 2002 p.285). A brief look at some middleware technologies is as detailed below.

2.4.1 Middleware Technologies

CORBA (Common Object Request Broker Architecture) is achieved by communication and linkage amongst 3 major components. These are the Object Request Broker (ORB), Object Adapters, and the Interface Definition Language (IDL). Some of the minor components that also interact with the 3 major in the architecture are interface repository, language mappings, stubs and skeletons, and dynamic invocations.

The ORB delivers requests to objects and makes sure that responses to the clients are implemented. It hides the different objects' location, implementation, execution state, and communication mechanisms. Through that, a developer specifies a target object (server) by use of an object reference (Moreira, R. et al., 2005). In addition, before the client makes a request, it must first know the operations and types that the target object supports. This is made possible through the Interface Definition Language which consists of definitions of different interfaces (equivalent to C++/Java classes) for objects (Moreira, R. et al., 2005).

While CORBA uses a repository for discovery of objects and their services, Enterprise Java Beans (EJB) have an abstraction entity called a bean context which groups a set of related beans in a hierarchy of contexts. It's also through it that dynamic addition of discoverable services is done. Service providers register and discover services on the context.

EJB defines a component model through standardisation of contracts and services offered by the runtime environment. It has 3 types of beans the use of which determine the developer's implementation of pre-determined interfaces. While CORBA relies on an open standard, EJB relies on Remote Method Invocation (RMI) which isn't an open standard although it supports objects passed by value and provides for dynamic class loading as well as execution of remote exceptions. EJB relies on RMI for transparent distribution of components' functionality (Moreira, R. et al., 2005).

The .NET framework has 2 main components i.e. the class library and the common language runtime. The class library holds pre-written classes used to develop applications on windows platforms while the CLR provides a run time environment for execution.

It converts intermediate language into native language which can run on heterogeneous systems through the help of a common type system (CTS) and a common language specification (CLS). The CTS determines declaration and use of data types during runtime execution to achieve interoperability of programming languages (Khan, S. & Hussain, W., 2008).

Due to .NET's dependency on Windows operating system and the current point of failure state of CORBA's ORB core, EJB would be preferred for it can run on any platform in addition to it being portable, scalable, and dynamic. 'D-Mash' hence seeks to implement object-oriented middleware through the use of java beans to help retrieve and update database values.

2.5 Integration

Gulledge (2006) details point to point integration as the developments of interfaces between different systems where the data model of the data source and destination is known by system integrators who determine how information is exchanged from one system to the other.

As Surugiu (2012) explains, an integration architecture based on web services is one of the strong methods used to integrate complex business processes. Different organisations with different business processes and workflow are able to exchange information through the web methods architecture.

Hashemi and Hashemi (2012) explain that one of the important goals of the service-oriented architecture as a solution to data type heterogeneity between systems is to improve interaction between organisations. This architecture is loosely-coupled enabling services running at run-time in different locations change their internal methods and procedures of executing various tasks.

The service-oriented architecture enables easy customisation and upgrading of existing applications which has led to rapid solutions to integration problems faced by organisations (Kester, 2013). It allows different applications to exchange data and participate in business processes regardless of the operating systems, programming languages, and diverse technologies underlying those applications (Kester, 2013). 'D-Mash' seeks to adopt this architecture by using web services and methods to ensure exchange of data between different online booking services provided by different organisations and the distributed system.

2.6 Previous efforts

Gulledge (2006) details point to point integration as the developments of interfaces between different systems where the data model of the data source and destination is known by system integrators who determine how information is exchanged from one system to the other. As Surugiu (2012) explains, an integration architecture based on web services is one of the strong methods used to integrate complex business processes.

Different organisations with different business processes and workflow are able to exchange information through the web methods architecture. In addition, an implementation exists that is used as an alternative where individuals book online through self-service. A construct of a bus with available seats exists that allows one to book online and pay using MPESA (a service used to transfer money as well as buy goods and services electronically using mobile phones).

This works for customers who have mobile phones with chip cards registered with mobile service providers like Safaricom and who have registered as users of the money transfer services. The service has to be up and working for the customer to do payment. Some officers in some company's bus stations serve the customers by following an un-automated procedure that entails use of mobile phones for communication amongst themselves to ensure that a bus' seat isn't reserved and booked twice. It's during such a process that the officer still ends up booking an already booked or reserved seat. This is because of using a manual procedure in the booking process.

2.7 Conceptual Framework

A conceptual framework is 'the system of concepts, assumptions, expectations, beliefs and theories that supports and informs your research' (Maxwell, 1996, p.25 cited in Leshem and Trafford, 2007, p.98). Most of the manual systems make use of asynchrony and centralisation as depicted in the model we've shown in Fig.1.

The model below is a conceptualisation of the efforts that are currently implemented by some entities in order to improve on business processes.

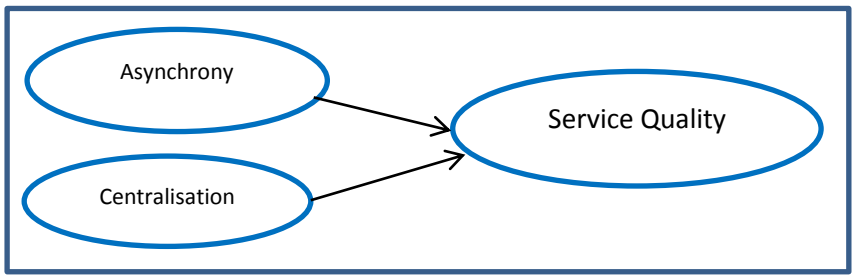


Fig.1 Current Model

The above model represents the current state where entities try to improve service quality on the basis of asynchrony and centralisation. However, it is imperative that a different alternative is developed based on the conceptual framework we propose in Fig.2 below due to challenges such as presence of single points of failure and bottlenecks.

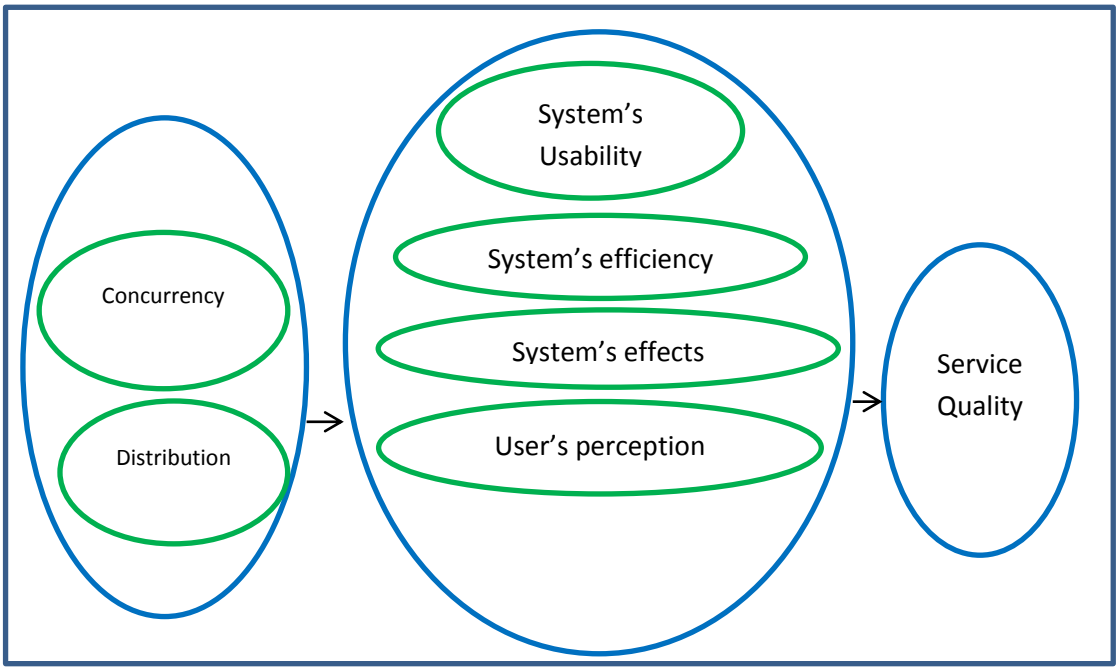


Fig.2 Conceptual Framework

The above framework represents a conceptualisation of the proposed state where entities try to improve on service quality on the basis of synchrony/concurrency and distribution. Further, intervening variables such as system's usability, system's efficiency, system's effects, and user's perception have been introduced.

The dependent variable is service quality while the independent variables are concurrency and distribution. In order to make use of this framework we hereby introduce its components as shown in the figure below.

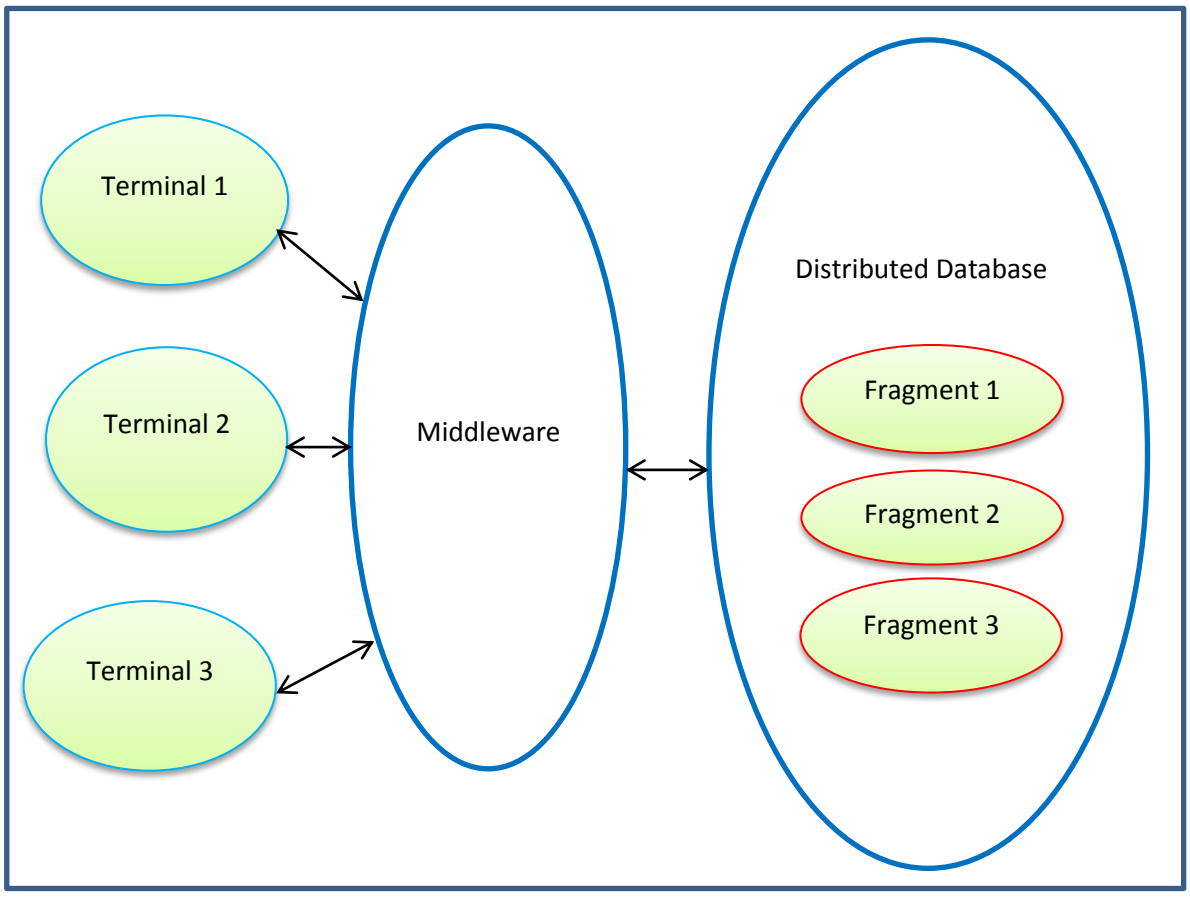


Fig. 3 Component model

As shown in the figure above, we operationalize the conceptual framework by having components i.e. a middleware, a distributed database, and distributed terminals. The middleware acts as the linkage between the distributed terminals and the distributed database. The distributed terminals have distributed interfaces that collaborate with the middleware to have their requests serviced.

The middleware receives their requests and immediately liaises with the distributed databases to determine the database to which the service request should be sent and worked on. It then awaits a response from the database through an interface that's implemented between the middleware and distributed databases.

Chapter Three

3.0 RESEARCH METHODOLOGY

3.1 Research Design

An action research approach was used to conduct the research seeking to fulfil two goals in the study them being contribution to knowledge and real-world problem solving. These were fulfilled by having the concurrency and distribution techniques, put into practice with an aim of testing their applicability in the real world as a solution to a problem encountered in a bus booking context.

It is through this that either the techniques' strengths or weaknesses were showcased. The strengths outweighing the weaknesses implies adoption of the techniques while the opposite implies further research to be made to come up with better techniques.

To achieve the goals mentioned above, an action research approach was preferred. It's appropriate for the project as it seeks to determine the applicability of proposed concurrency and distribution techniques hence practice informing research and research informing practice by having the two complementing each other synergistically as explained by Avison et al (1999).

3.1.1 Why Action Research

It is the preferred approach here for it "combines theory and practice through change and reflection in an immediate problematic situation within a mutually acceptable ethical framework" (Avison et al., 1999). A system, which is implemented through the use of new techniques, is introduced as the change seeking to solve the concurrent bus booking problem in a distributed environment, then reflection on its strengths and weaknesses is done.

Avison et al. (1999) further details that action research is a process consisting of one or more iterations that involve researchers and practitioners working together to diagnose the problem, intervene with an action, and perform reflective learning on it. As McKay and Marshall (2001) elucidate, an action researcher has 2 aims that lead to the action research approach having two cycles overlaying on each other. The cycles are illustrated in the following diagrams.

The first cycle relates to the researcher's research interests and responsibilities relates to the researcher's problem solving aim while the second cycle relates to the researcher's problem solving aim (McKay and Marshall, 2001). The figure below represents the researcher's research interests.

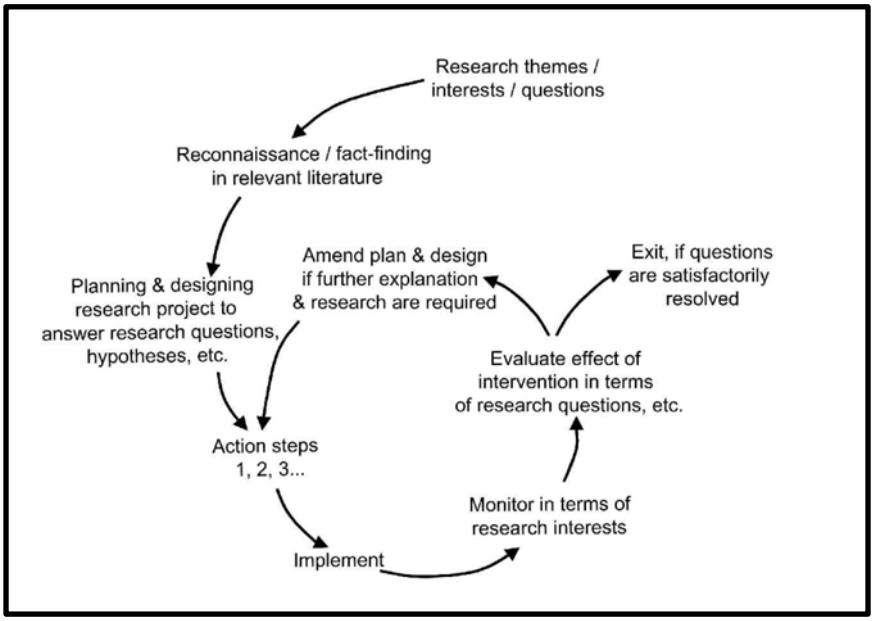


Fig.4 'The research interest in action research' (McKay and Marshall, 2001: p.50)

As the figure above shows, the researcher has some research interests / themes / questions which is followed by fact-finding in relevant literature after which planning and designing of the research project towards the theme is done. The main theme in this project is to assess the impact of automated concurrency and distribution on service quality. Iteration then follows that seeks to satisfactorily resolve the research questions / themes.

The cycle below represents problem solving as one of the aims of an action researcher. As demonstrated, there are 9 steps one of them being part of an iteration. This step is amendment of the plan if further change is desirable.

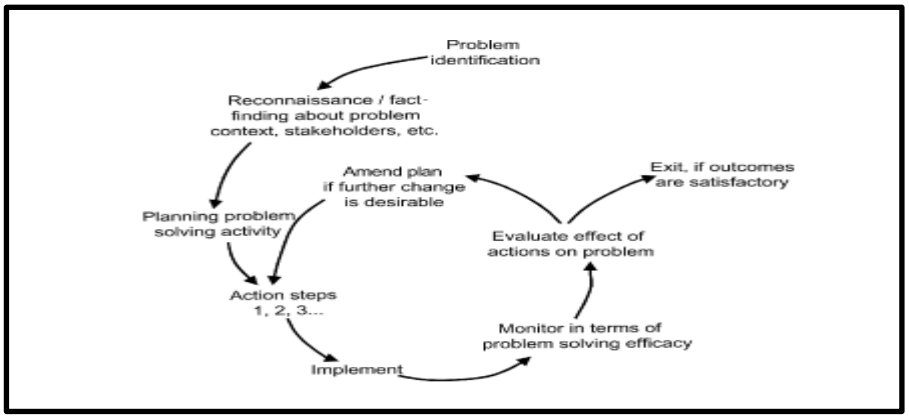


Fig.5 'The problem solving interest in action research' (McKay and Marshall, 2001: p.51)

The figure below shows a juxtaposition of the 2 aims in one diagram. It shows an action researcher's research interests and responsibilities besides the problem solving interest.

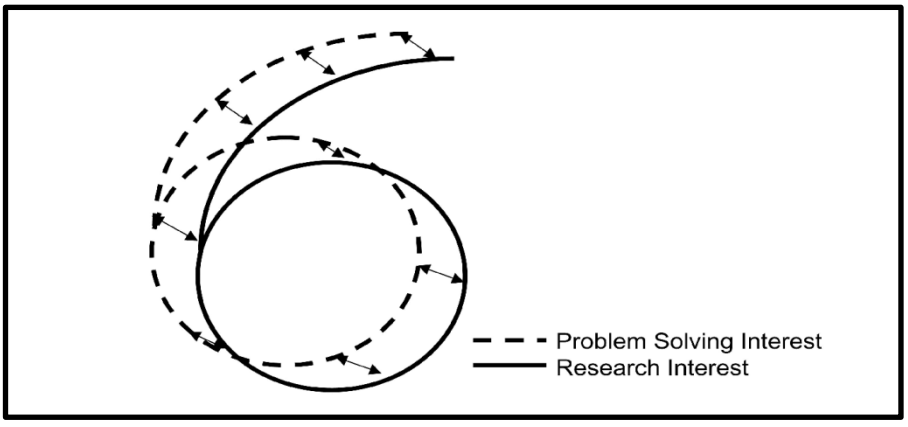


Fig.6 'Action research viewed as a dual cycle process' (McKay and Marshall, 2001: p.52)

As shown above, action research involves a dual cycle process. This process entails iterations of problem solving combined with research interests in order to help an action researcher contribute to knowledge and society.

3.1.2 Strategy

The preceding diagrams are an illustration of the 2 aims of the common cycle, which is shown in the following figures, of action research. This cycle is composed of 5 phases namely diagnosing, action planning, action taking, evaluating, and specify learning.

Fig.7 'Representations of the action research cycle' (McKay and Marshall, 2001: p.49)

As shown in the diagram above, the development of a client-system infrastructure entails an action researcher meeting a client's needs and requirements during the 5 phases which are all working towards one goal.

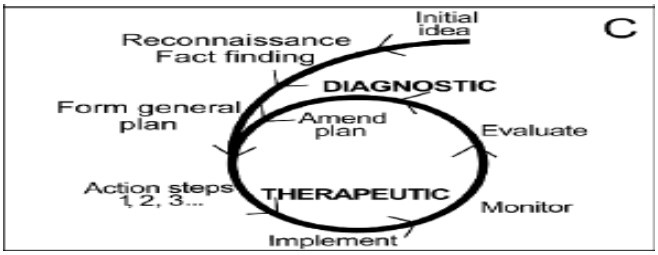


Fig.8 'Representations of the action research cycle' (McKay and Marshall, 2001: p.49)

Figure 8 above shows that an action researcher's common cycle entails a diagnostic period and a therapeutic phase which involves introduction of an action as a therapy towards problem solving.

It's during the diagnosing stage that identification of the main problem calling for the change is done. This is then followed by the action planning phase where actions intended to bring about a desired future state better than the current problematic state are discovered. Subsequent to that step is action taking which entails researchers-practitioners collaborative implementation of the actions planned to bring about changes that are expected to have some effect (Baskerville 1997, cited in Baskerville 1999, p.15).

Evaluation of the outcomes follows where the effects of the action taken are analysed. If a positive effect, it must be keenly looked into before a conclusion is made that the action, introducing the change, solely caused it. If negative, a new framework for the next iteration in the action research cycle should be established. The specifying learning activity then follows, upon evaluation completion, where the success or failure of the framework undergoes analysis and critique that contributes to knowledge for researchers to use in their future research (Baskerville 1997, cited in Baskerville 1999, p.15).

McKay and Marshall (2001) also assert that a reflection is also made on, the problem situation in which the researcher is interested and which can be solved using a particular practical problem solving method, on action research as the research method, on the problem situation in which the organisation is interested, and on the practical problem solving method hence leading to what is called experiential learning or learning from doing (McKay and Marshall, 2001).

In line with the above, it was collaboratively identified, as part of diagnosing, that there was a manual process, followed by officers located in different locations, through the use of mobile phones to concurrently book seats, that led to some instances of 1 seat's allocation to more than 1 passenger hence concurrency in a distributed environment still being a problem. The organisation desired to have a situation where no seat can ever be allocated to more than 1 passenger.

In order to achieve this, a concurrent distributed system based on a conceptual framework that seeks to achieve concurrency in a distributed environment, was introduced as an action to bring about change expected to have an effect. This effect was analysed during the evaluation phase to help in subsequent determination of the

success or failure of the framework serving as the basis of the system. In addition, as detailed later in Chapter 4, reflection was also made, on the concurrency problem in a distributed environment, on the research method used, and on the practical problem solving method used. The realisation of the research method's and framework's fate then serves as contribution to knowledge and as a basis for the next framework to be used by researchers and practitioners in future problem-solving and research endeavours.

3.1.3 Philosophical approach

Although action research is mostly done with a critical theorist's ontological assumption, this research is based on a pragmatist approach which is a combination of positivist, constructivist, and critical approaches. As a positivist, a concern exists of comparing the status before and after the action was implemented while as a constructivist, a concern exists of the kind of views and perceptions participants have towards the change process (Easterbrook et al. 2008).

From a critical theorist's point of view, it is taken that a real problem exists that needs to be solved, the adopted solution is desirable, and that emphasis is laid on the lessons learnt that help other researchers who wish to pursue research in the same field (Easterbrook et al. 2008).

3.1.4 Methodology limitations / Validity threats

There are 3 common threats that have been identified with action research namely uncontrollability, contingency, and subjectivity (Kock, 2004). The uncontrollability threat refers to where the researcher has incomplete control over the research environment (Kock, 2004). As Avison et al. (2001) cited in Kock (2004, p.268) explains, rarely will an organisation yield complete authority to an external researcher.

This is because of the fact that an action researcher has dual goals as explained earlier. Kock (2004) adds that action research faces the contingency threat where contingency is synonymous to the difficulty of generalising research findings in contexts different from the one in which the finding was discovered. This can be an issue for different constructs are usually used by action researchers dynamically

depending on the type of problem and research cycle's outcome. The subjectivity threat rides on the fact that the personal involvement of an action researcher is likely to force her interpret the findings in some ways leading to some incorrect conclusions (Kock, 2004).

It's inherent that deep personal involvement of an action researcher has the potential to bias results for it is impossible for the researcher to be in a detached position while at the same time introducing a positive intervention (Kock, 2004). The likelihood of the manifestation of this threat is high because the nature of action research requires one to unavoidably engage client organisations in the research process.

3.1.5 Validity threats' remedies

In order to counter the threats above, we sought to implement strategies them being, use of units of analysis, grounded theory, and iterations. Units of analysis are the entities whose reactions or behaviour are probed to help in generalisation. The more the analysis of several instances of these units in different contexts, the higher the external validity (Kock, 2004).

Grounded theory is mainly used to work against the subjectivity threat by employing a coding process that promotes objective data analysis besides ensuring that different coders will produce the same results (Kock, 2004). The coding process entails identification of variables, links between the variables, and dependent variables to which certain effects are associated (Kock, 2004). The variables identified were system's efficiency, system's impact, service quality amongst others. This helps the researchers have an easy time when tracing back to present the facts serving as a basis of theories and models.

Iterations are usually used to collect cumulative data about units of analysis in different contexts hence being able to develop evidence gathered from different iterations in the action research cycle (Kock, 2004). In case problems occur, outside the control of the researcher, and that affect data collection and analysis, the researcher always has an option of going through another iteration hence neutralising the uncontrollability threat (Kock, 2004). In addition, the iterations help in expanding the research scope with an aim of improving on the generality and

external validity through identification of invariable patterns hence counteracting the contingency threat (Kock, 2004). This came out as a suitable method we sought to use in order to limit the presence of bias because of its allowance for iterations until the objectives of the research are met. Uncontrollability was countered in this project by seeking to terminate or halt the action research cycle in case something happened beyond our control after which a solution to the hindrance would first be sought for.

We sought to encounter contingency by having units of analysis from the beginning. In addition, some that are important or closely related to the first ones were identified and examined in the course of the research process. Examples of these units are the system, users, and the framework. We sought to encounter subjectivity by having variables and seeking to find out the relationships between them.

These were system's efficiency, system's impact/effect, system's usability, users' perceptions towards the new system, and service quality. It was expected that successful application of the framework would positively affect efficiency, impact, usability, and users' perceptions which would in turn positively contribute to service quality. Identification and use of these variables helped us focus on the objectives without letting personal or emotional interference that usually have the likelihood of introducing bias.

3.1.6 Tools, procedures, data collection, and data analysis.

Overall, we collaborated with a Bus Company which in this case represents a sample from the large population set of bus companies in Kenya. However, a further sample was drawn within the several Bus Company's branches countrywide. Therefore, the target was the company's employees from Nairobi head office branch approximated as a sample size of 8 booking officers. As part of action planning, requirements analysis was done using prototyping and open-ended interviews while action taking entailed using a continuously developed and modified prototype as the opinions, requirements, and suggestions were received from the users of the system.

In addition, integrated system testing towards the end was done to ensure that the system worked as expected. During the full prototype evaluation, the users were able to test the system and give qualitative and quantitative feedback on their interaction with it and on how it eased their work.

As part of evaluation, the feedback was looked at and analysed to determine the kind of effect the framework and system had on the business process of the organisation and on the problem as a whole. This feedback and analysis is available in Chapter 3 and 4. In addition, the system acted as an instrument in aiding the research. Participatory observations and interviews were conducted to understand the communication modes used in the current booking process as well as the challenges involved and at the end during evaluation.

We went for interviews, participatory observations, data archival, and set-up of a quasi-experiment as our data collection methods to enable in compilation of quantitative and qualitative data from the sample. Data analysis was done by performing statistical analysis, with the help of PSPP & Microsoft Excel, and coding done on the textual data collected during the interviews. Qualitative and quantitative analysis methods were therefore used to complement each other and to achieve triangulation that improves on the validity of the research.

These data collection and data analysis methods came out as the best for this kind of research because the sample was of a small size on the scale of large, medium and small. It's also through them that both qualitative and quantitative information regarding the system's variables and framework was found and analysed on top of them being common good methods used to counter research validity threats.

They also go well with the action research approach that allows for development of practical solutions to real-world problems using software development tools, principals, and techniques. It's during the specifying learning stage, as later detailed in Chapter 4, that the outcomes of evaluation were reflected on where a positive outcome, about the solution to the concurrency problem, and the applicability of the research and problem solving methods, was expected.

It was also expected that this outcome would consequently serve as contribution to knowledge and as a basis for the next framework to be used in future problem-solving and research endeavours. Besides that, the system was expected to lead to service quality improvement that in turn would lead to an overall economic growth.

Challenges, recommendations, and conclusions were documented and published, as later detailed in Chapter 5, to aid in future research. The techniques and methods used in the software development process as well as any changes made on the same to achieve the research project's goal were also documented. It was also expected that this would contribute to the distributed computing sub-field.

3.2 System Analysis

This section gives a description of the current and proposed systems as part of action planning and action taking.

3.2.1 Description of the old system

The figure below represents a data flow diagram of the old system.

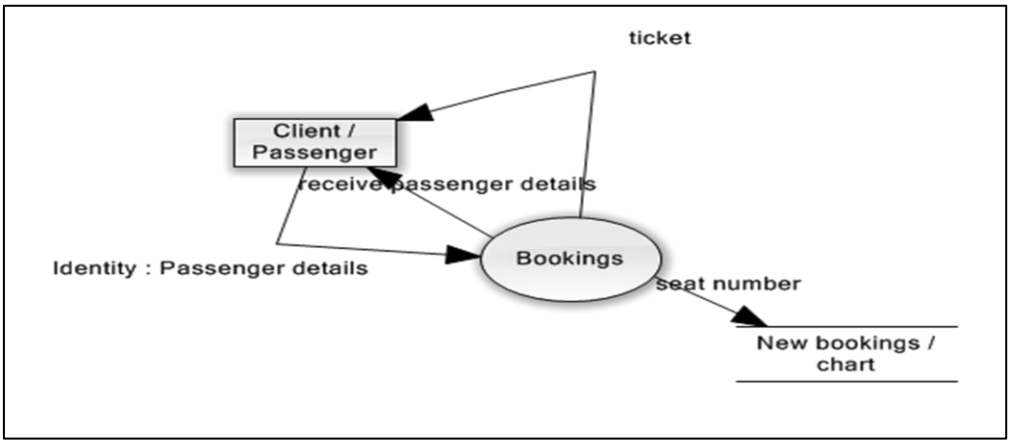


Figure 9: Old System DFD

The data flow diagram illustrates the flow of documents (physical aspect of the DFD) and the flow of data (logical aspect of the DFD) into and around the old manual booking system. The customer provides identification with his/ her details included. These are received by the booking officer who then records the details and books the seat in a bus that the customer prefers. The client then receives a ticket with all details included after payment is confirmed.

3.2.2 Description of the proposed system

The figure below represents a data flow diagram of the proposed system.

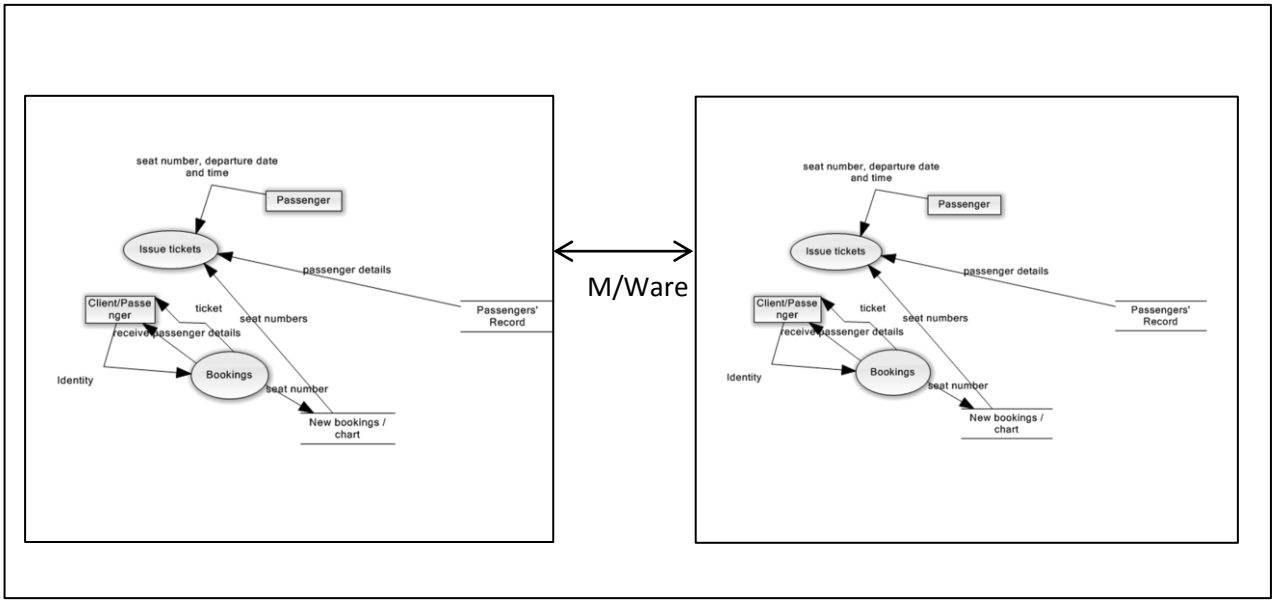


Figure 10: Proposed System DFD

The data flow diagram illustrates the flow of documents (physical aspect of the DFD) and the flow of data (logical aspect of the DFD) into and around the proposed booking system. The customer identifies himself / herself by providing an identification document. His / her details are received by the booking officer who then records the details and books the seat in a bus that the customer chooses.

The client is provided with a ticket once payment is confirmed. In addition, the customer can confirm his / her booking details upon losing a ticket by providing the seat number, departure date, and departure time. Besides that, more than 1 booking officer can concurrently book the same seats in the same buses from different locations without conflict.

3.2.3 Functional Requirements

To achieve concurrency and distribution in a proposed system as shown above, several functional requirements need to be met. As part of **action taking**, a prototype should exhibit the requirements shown below.

i) The system should enable booking officers concurrently book seats without any conflict.
ii) The system should be able to allow different booking officers in different locations successfully book seats for the customers.
iii) The system should be able to provide details of all customers who have booked seats.
iv) The system should be able to prevent double booking of seats.
v) It should be able to generate tickets for the customers.
vi) The system should only allow registered booking officers to interact with and use it in provision of booking services.
vii) It should allow remote booking without any officer knowing the location of the remote destination. It should appear as if the booking was done in that station where the officer is located.
viii) It should enable retrieval of customer details in bookings done by different officers in booking offices.

3.2.4 Non-Functional Requirements

To achieve concurrency and distribution in a proposed system as shown above, several non-functional requirements need to be met. These are:

Reliability

The system should work for a long period of time without failing i.e. a period of 364 days with a failing allowance of 1 day in a year.

Availability

The system should be present and working as expected whenever needed in the course of the year. i.e. 7days a week during the year whether leap or not.

Security

A username and password should be availed so as to allow a booking officer to log in and use the system.

Maintainability

It should be possible to routinely backup the databases after every close of the day's business so as to help in future retrieval for ticket data or bonus programmes.

3.3 System Design

As part of action planning and action taking, this section details the system architecture and distributed database design.

3.3.1 System Architecture

Distribution is important in today's systems for it helps in distribution of several elements e.g. processing logic instead of overloading one machine with a particular task. Function is also an important element that needs to be distributed in addition to control and data. Today's organisations are distributed in nature with several branches country-wide, region-wide, and world-wide.

Such enterprise models are reliable and more responsive to customer's needs. Many of the current applications are also distributed e.g. ecommerce applications over the internet. To solve some of the communication and data management problems that come with such models, distributed processing has to be achieved.

Figure 11 is a distributed database management system's *architecture* with a distributed database (a collection of multiple, logically interrelated databases distributed over a computer network) whose data has been physically distributed and stored in several computers communicating over a network. In distributed systems,

hardware and software are heterogeneous. The database is also distributed over the distributed nodes connected using a computer network. In addition, the middleware communicates with the distributed peers through web-services that have been implemented at the middleware and peer levels.

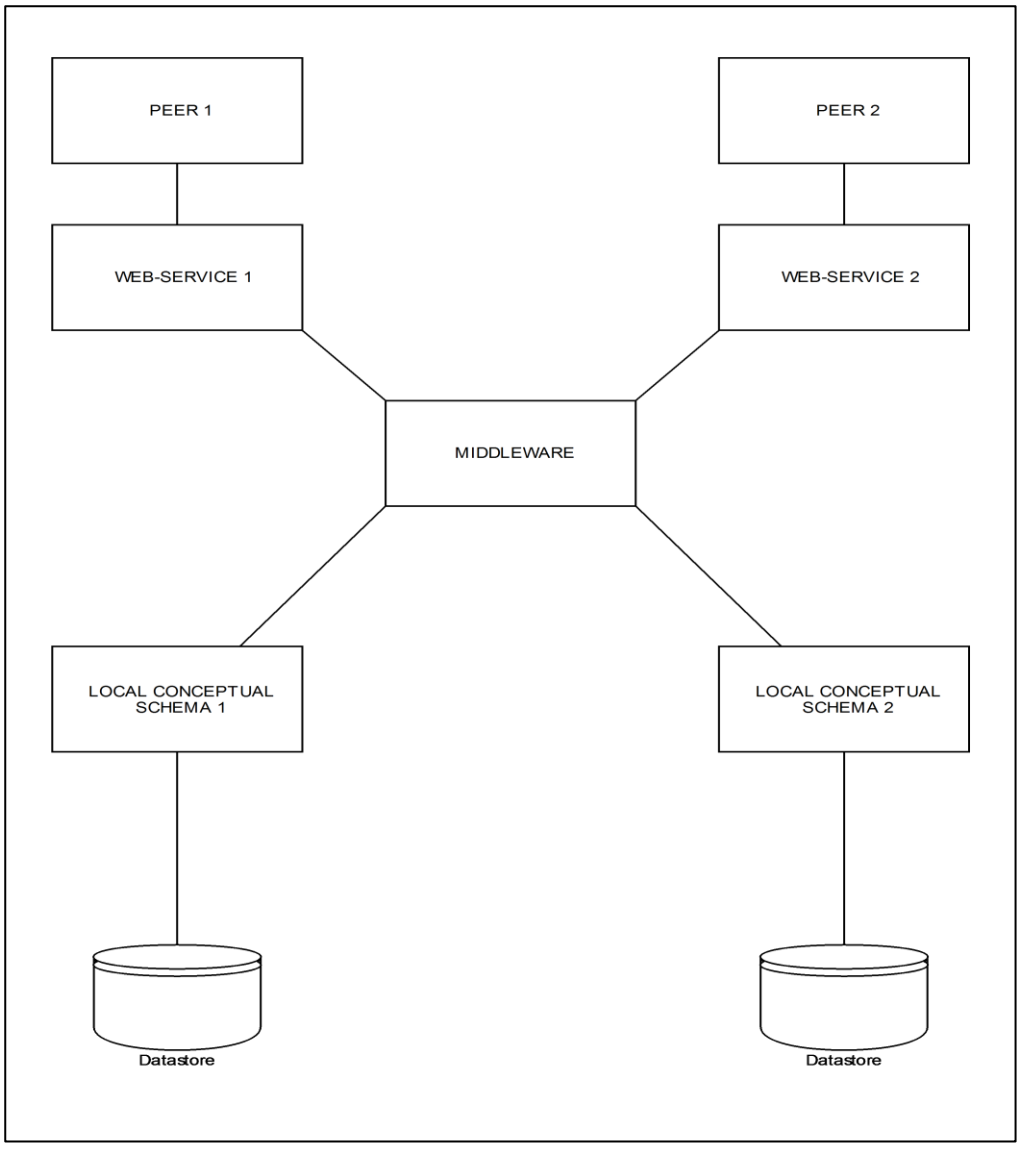


Figure 11: System's Architecture

The figure above represents the distributed database management system architecture. Every terminal represents a site/branch/booking station with its own autonomy and heterogeneity in the sense of its implementation being of different technologies compared to its peers'. The physical data organisation on each Data

store may be different in that it could be using PostgreSQL's data organisation structures while the other could be using MySQL's.The global conceptual schema describes the logical structure of all the sites. It's a combination of all the local conceptual schemas. The local conceptual schema is important for it helps in handling data fragmentation and replication. These schemas provide location, replication, and network transparencies. The user successfully queries data irrespective of its location and irrespective of which local component/schema/data store will service it.

Global queries are translated into a group of local queries directed to the respective data stores. User applications/interfaces access the local data via the external schemas which directly interact with or subject their queries to the global conceptual schema through which subsequent routing (with the help of a *middleware*) to the respective local conceptual schema representing each local data store/site is done.

3.3.2 Distributed Database Design

Whilst the data in a distributed database is spread across the nodes, some of it needs to be fragmented and some needs to be replicated such that replicas of it exist in the distributed sites. This is after it is known that this type of data is mostly needed so that it doesn't need to be stored at the hosting site only (where it would take time to pull and provide it to several users requesting for it) but replicas of the same to be stored at the other sites for quick retrieval. Fragmentation entails having data localised such that data about a particular entity is only stored at its local site rather than having it stored in the other distributed sites unnecessarily.

The distributed database is hence fragmented and replicated/duplicated and with transparency in that a user can pose a query and interact with it faster via a distributed system's interface without wanting to know the location, fragmentation, and replication of the data but instead lets the system resolve that. Most of the important data considered to be important and frequently needed is hence replicated in several sites such that single points of failure are eliminated for the replicated data

will still be available hence the entire system isn't brought down. This strategy is used besides having recovery and backup procedures in place. In addition, distributed transaction processing is implemented to ensure that business goes on even when un-replicated data is not available when the host database is not available due to network, server, or other failures. This type of processing enables concurrent execution and synchronisation of distributed transactions such that proper and accurate updates are made to the host database upon resumption.

Distributed concurrency, reliability, replica control, and recovery protocols are enforced to achieve this. The distributed database consists of 2 distributed database nodes / 2 horizontal fragments and 2 replicas (1 replica present on both sides). The individual local schemas on every node communicate with one another through the help of a middleware that determines where to commit a transaction or not.

3.4 System Implementation

This section details the implementation of the system. It explains the tasks undertaken to deploy the system.

3.4.1 System set-up

- Installation of 2 MySQL distributed database nodes on 2 different locations. Each database node has 4 tables. To access the database Apache and MySQL interfaces are used instead of the console.
- Install the Java Development Kit in 2 servers that host the distributed database. The kit provides a Java Virtual Machine that enables running of java programs.
- Install NetBeans Integrated Development Environment with Glassfish application server embedded within. The application server provides a deployment platform for the web prototype.
- Set up a network connection to the servers ensuring that the Internet Protocol addresses to the server never change.

- Perform all configurations and deploy the client applications in the booking offices as well as the middleware application that helps in retrieval and committing of transactions in the right database nodes.

3.4.2 Clients' JSF interfaces and middleware invocation classes

JSF Log-in Interface

Below is a screen shot of the booking officers' log-in page followed by the mark-up language used to implement it

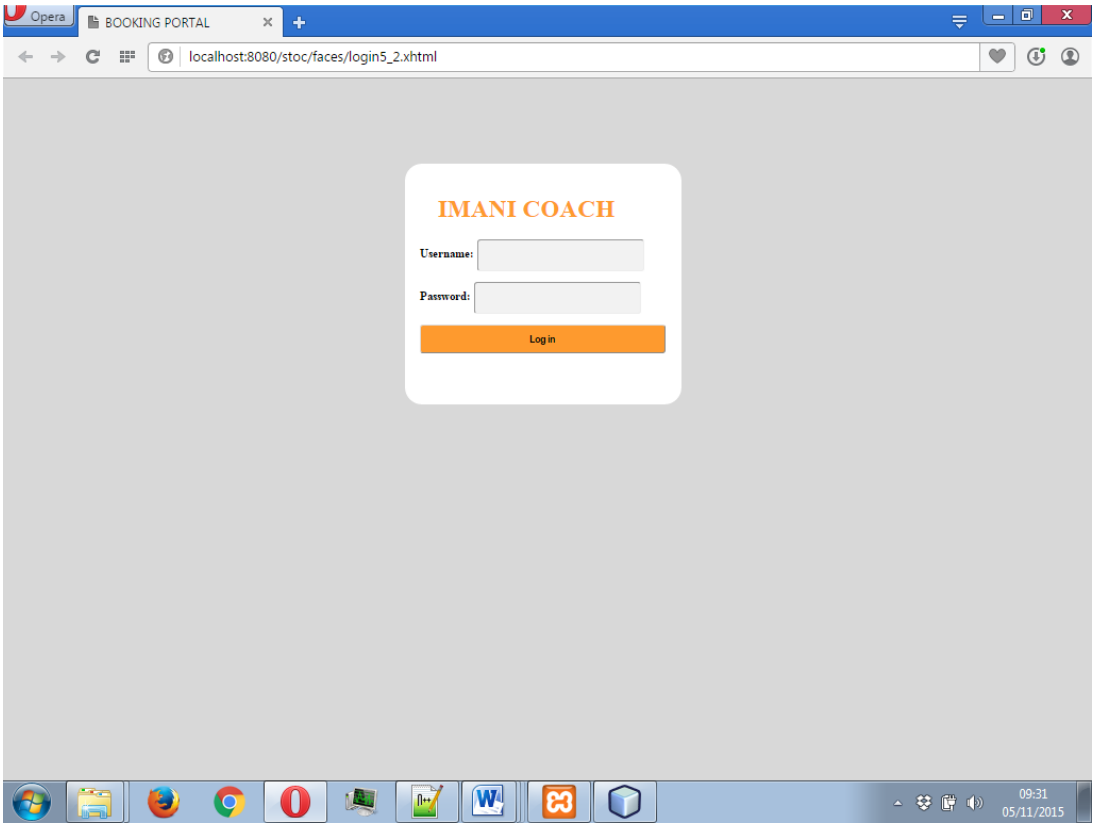


Figure 12: Prototype's log-in page

The code for the log-in page above can be found in the appendices section.

JSF Booking Interface

Below is a screen shot of the booking page followed by the class invoked upon clicking on the book button.

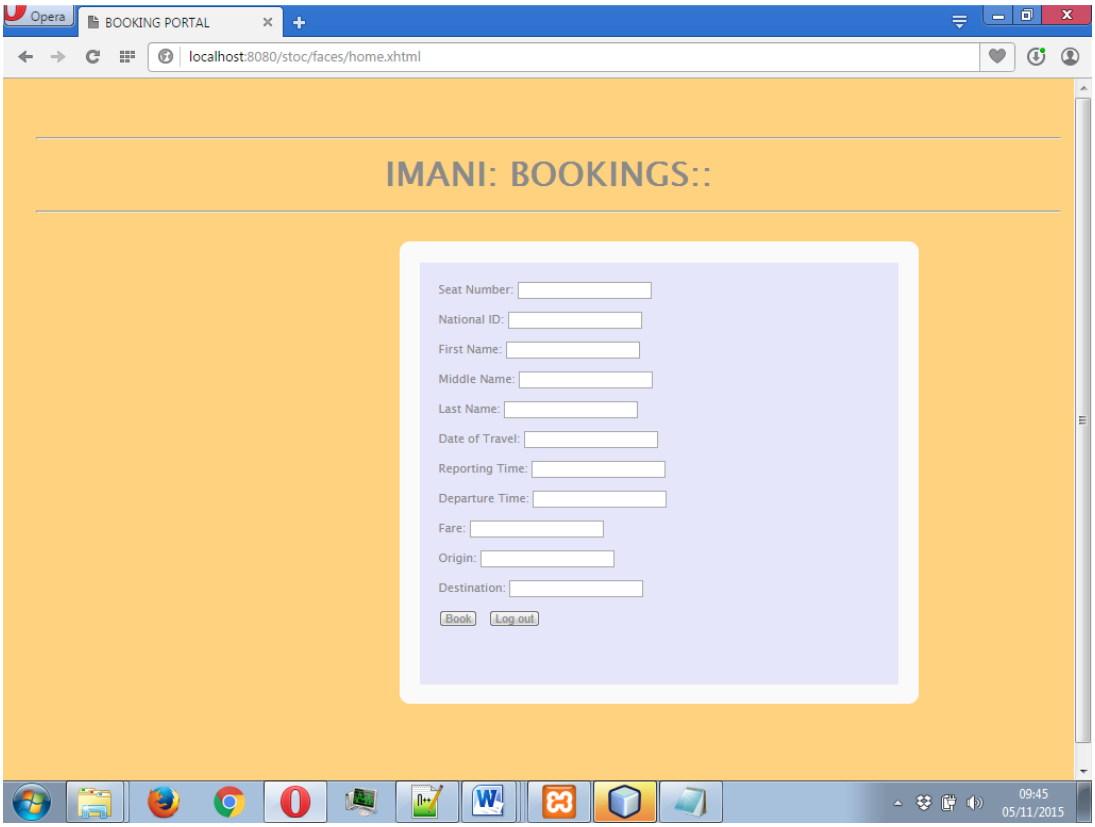


Figure 13: Prototype's booking page

The interface above has been implemented using Java Server Faces and Extended Hypertext Mark-up Language (XHTML). A booking officer fills the form and clicks on book after which a client method is invoked. The method proceeds to remotely invoke a middleware method that determines which distributed database node is to be affected. The middleware class is invoked through a middleware's web service implementation. A representation of the Java class representing it is shown in the code in the appendices section. The concurrent distributed prototype was implemented using Java and a MySQL distributed database. It was deployed upon setup of internet connection from an Internet Service Provider.

3.5 Data Collection

This section entails collection of data regarding the performance of prototype and the effects resulting from its use. A quasi-experiment was employed to help in collection of data during evaluation of the prototype so as to help in analysing the kind of contribution it has towards service quality. This method was chosen for there was no significant randomisation exhibited.

In addition, resources were not enough to enable set-up of two separate experimental groups of officers in 2 offices instead set-up was done using one group where the treatment / action was introduced as the parallel changeover method was employed. Comparison was done as booking of every seat happened manually and using the system.

Also, archived data and participatory observation methods were used to complement the experiment in data collection. This research project involved a quasi-experiment that was of one group pre-test and post-test design type and that entailed evaluation before and after introduction of the action.

3.5.1 Pre-test data collection

As part of pre-test, interviewing and data archival methods were employed to collect quantitative and qualitative data. It was found out from the interviews that **2** to **3** customers complained of double-booking per month and that double-booking would happen **frequently** especially when there is demand of the service.

It was also found out from past archived records that an average of 44 seats was booked from a sample of 2543 seats that were booked in 58days. Amongst the 58days, a 10-day sample together with its descriptives is illustrated using a column bar graph as shown below. It illustrates the seats that were booked within the 10 days and the probability for double-booking.

The figures below represent a sample of the pre-test phase where a total of 405 seats were booked and a double-booking scenario of 3 seats.

DAY	SEATS	DOUBLE-BOOKING
ONE	41	1
TWO	39	0
THREE	39	0
FOUR	41	1
FIVE	45	0
SIX	48	0
SEVEN	22	1
EIGHT	47	0
NINE	41	0
TEN	42	0

Table 1.1: Pre-test sample data (double-booking chance of 2-3 seats)

The figure below shows the statistical descriptives of the pre-test data shown above where the minimum number of seats booked was 22 while the maximum number of seats booked was 48.

DAY	SEATS	DOUBLE-BOOKING
ONE	41	1
TWO	39	0
THREE	39	0
FOUR	41	1
FIVE	45	0
SIX	48	0
SEVEN	22	1
EIGHT	47	0
NINE	41	0
TEN	42	0

Table 1.2: Pre-test descriptives

Additionally, the range is 26 while the median is 41 seats. Further, the following figure shows the plotting of the data using a column bar graph.

As shown in the diagram below, the seats that haven't been double-booked are represented using the blue colour while the double-booked seats have been represented using the light brown colour.

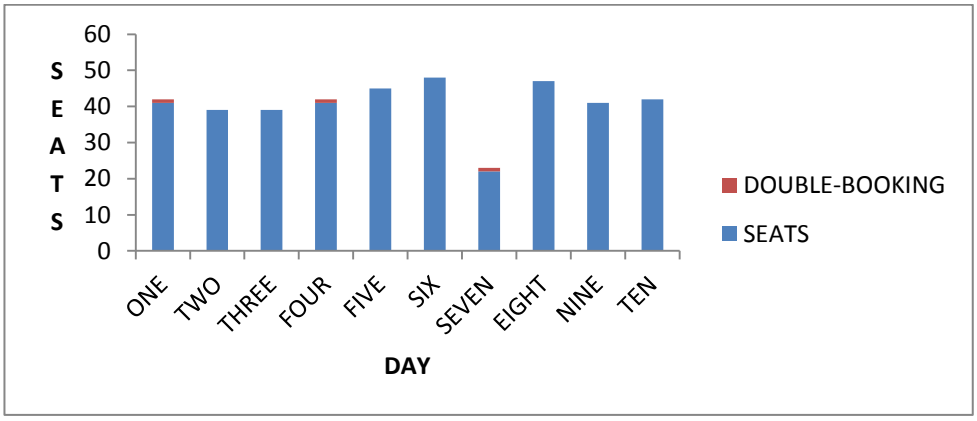


Figure 14: Pre-test distribution (double-booking chance of 2-3 seats)

As informed in the interview, the probability of 2 to 3 out of 1257 seats (per month) being double booked, due to effective concurrency and distribution not being met, can also be illustrated as shown in the figure below.

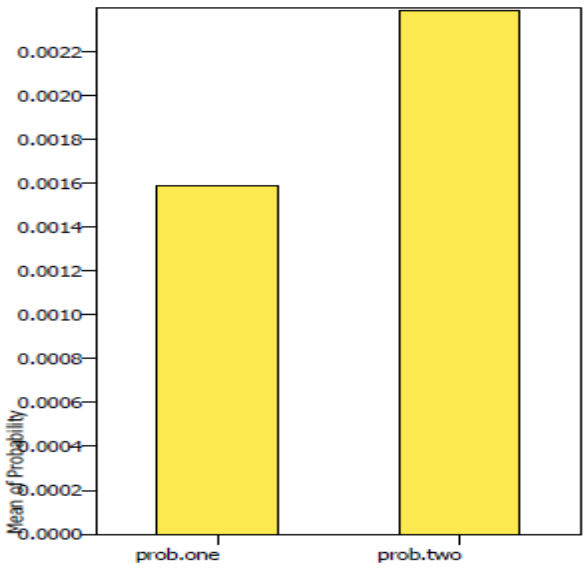


Figure 16: Pre-test probabilities (double-booking chance of 2 – 3 seats)

Some of the questions asked during the pre-test interview were:-

- "What are your opinions about double-booking and how do you feel about it?"
- "Can you describe the image that's usually portrayed once it happens?"
- "Does the double-booking scenario happen intentionally? Please explain."
- "How easy or hard is it to deal with this situation? Please explain. Have you tried to avoid it before? What do you do to try avoiding it and does it still occur even when trying to avoid it?

The questions above were answered simultaneously as follows:-

- "This is an occurrence that brings about serious problems. First and foremost you may reach a point of being taken to the police because of interfering with one's journey. Secondly, if he had booked his seat and it occurs that I had booked this seat to a different person it then means that he won't have a seat. In fact, the other person may have been the first to book the seat and it happens that the second person to book the same seat arrives earlier, than the first person who booked, and occupies it on the day of departure. The first person may take you to the police where you'll be forced to pay him damages for interfering with his/her journey for he had booked in advance. "
- "In the eyes of the customers, it appears that we are not qualified in our work. It's a bad image."
- "It happens unintentionally. Truly, it isn't a good thing."
- "It isn't easy to deal with the situation for the passenger is usually disappointed upon us abruptly interfering with his journey yet he/she had prepared himself/herself and booked the seat early enough. We try to avoid it by trying to be keen. However, it's usually difficult to maintain that level of keenness hence we find it happening even as we try to avoid it."

Participatory observation

From observation, it was found out that 1 client had no seat for it appeared that 10 seats were available yet the booking officer had been assured of 11 seats being available. However, the booking officer anticipated this from experience and helps the customer through reservation of an extra seat in case one finds his/her seat occupied by a different person.

3.5.2 Post-test data collection

As part of post-test, interviewing and data archival methods were employed to collect quantitative and qualitative data. It was found out from the interviews that **0** customers complained of double-booking within the testing period and that **0** seats were double-booked using the system.

It was also found out from the records that an average of 45 seats was successfully booked from a sample of 449 seats that were booked in 10days. The sample, together with its descriptives, is illustrated using a column bar graph as shown below. It illustrates the seats that were booked within the 10 days and the **0** probability for double-booking.

DAY	SEATS	DOUBLE-BOOKING
ONE	41	1
TWO	39	0
THREE	39	0
FOUR	41	1
FIVE	45	0
SIX	48	0
SEVEN	22	1
EIGHT	47	0
NINE	41	0
TEN	42	0

Table 1.3: Post-test sample data

As detailed in the table above, there was neither chance nor case of double-booking in the sample which had a total of 449 concurrently booked seats.

The figure below shows the statistical descriptives of the post-test data shown above where the minimum number of seats booked was 33 while the maximum number of seats concurrently booked using the system was 48.

DAY	SEATS	DOUBLE-BOOKING
ONE	41	1
TWO	39	0
THREE	39	0
FOUR	41	1
FIVE	45	0
SIX	48	0
SEVEN	22	1
EIGHT	47	0
NINE	41	0
TEN	42	0

Table 1.4: Post-test descriptives

As shown in the diagram below, the light-brown colour is not represented in the 10 days because there were 0 instances of double-booked seats. The other part of the column graph represents concurrently booked seats.

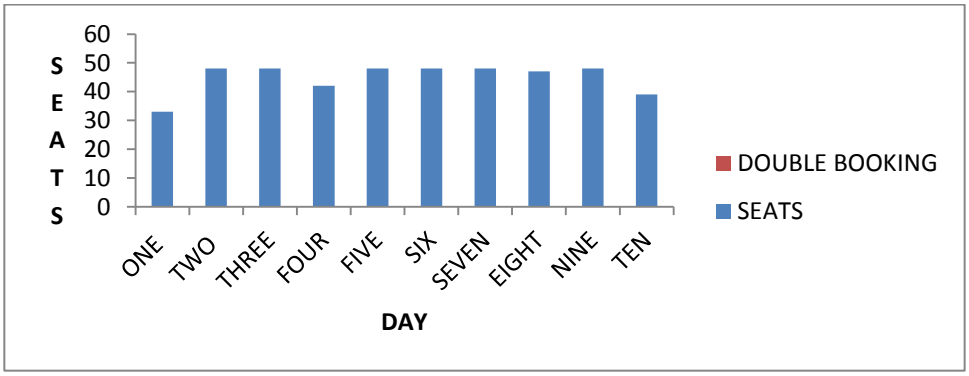


Figure 16: Post-test distribution

As informed in the post-test interview, a probability of **0** out of 315 seats (per week) being double booked, due to effective concurrency and distribution being met, has been illustrated in the following figure. This post-test probability 0.000000 (0/315) is hereby compared with one of the earlier pre-test probabilities of 0.001591 (2/1257).

The column graph below shows the change in double-booking probabilities before and after introduction of the prototype.

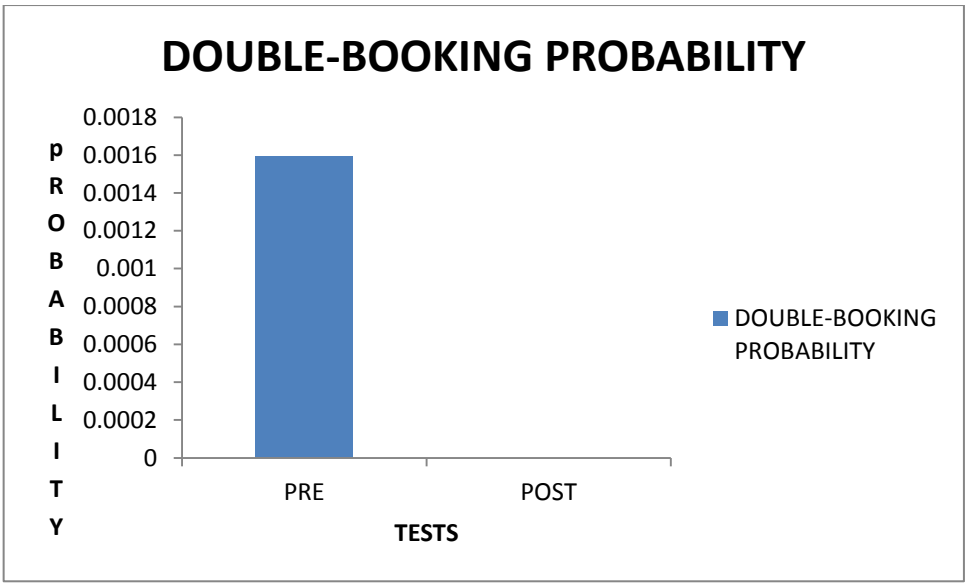


Figure 17: Pre and post-test probabilities

Some of the questions asked during the post-test interview were:-

- "Please describe your interaction with the system"
- "Would you recommend such a system to any other organisation and why?"
- "How possible is it for a customer to confirm a booking and how long does it take for you to confirm?"
- "What level of neatness and presentation is present if any and was it there before introduction of the system?"
- "How possible is it now for you to concurrently book a seat without any case of double-booking?"
- "Please explain how the system has affected your service to the customers"
- "Generally, how have you found the system? Has it helped you?"

The questions above were answered simultaneously as follows:-

- “The use of the system has been good. It’s okay and operates well. It has made my work easy.”
- “Yes, I would recommend it for use elsewhere. This is because we have tested it and found it to have been good.”
- “It is possible to confirm and takes 3 to 5 seconds.”
- “There is a high level of neatness and presentation as opposed to the previous manual system. The previous one had errors whilst the new one has improved on the order and presentation of our work.”
- “Yes, it’s possible. A seat will only be available for booking to the first person.”
- “My service has been improved for a client finds progress in our use of technology.”
- “It has been helpful for instance it has helped in booking and printing of customers details clearly on the ticket. It also alerts you in case of an error and has also helped prevent double-booking.”

Chapter Four

4.0 DISCUSSION

This chapter contains details on prototype evaluation and analysis as part of the evaluation phase as well as a look at the specifying learning phase where the results of the evaluation phase are discussed, reflected on, and the success or failure of the conceptual and practical frameworks analysed.

4.1 Prototype evaluation and data analysis

In line with the evaluation phase's tasks, this section entails analysis of the quantitative and qualitative data collected before and after introduction of the prototype to determine the kind of effects that the action brought about.

4.1.1 Pre-test analysis

It was found out that a probability existed of a double-booking scenario occurring which led to serious problems to a point of being possibly taken to the police and paying damages to the client. In addition, the existence of this probability meant a seat being made available to more than 1 person and consequentially led to the annoyance of customers hence portraying a bad image of the company.

It was found out that it wasn't something that occurs intentionally besides it being hard to deal with once it occurs. The officers would try their level best to be keen when booking but found themselves double-booking some seats even as they never intended and tried to avoid it.

It was also observed during the participatory observations that 1 customer had missed a seat even after the booking officer was earlier assured of its availability. This continued to expose the possibility of a seat being double-booked.

With the help of the bar chart plotting the pre-test probabilities, it comes out clearly that a probability of double-booking exists yet it shouldn't hence the pre-test results therefore generally suggest existence of a double-booking problem which occurs unintentionally and that comes with disgusting consequences. This is because of the booking officers trying to concurrently book a seat from different locations without use of distributed system software.

4.1.2 Post-test analysis

It was found out that zero or no probability existed of a double-booking scenario occurring. Comparing this outcome with the previous one, 0 seats were double-booked when using the prototype. In addition, the absence of this probability meant a seat being made available to 1 person only hence preventing double-booking.

It was found out that the system would prevent an officer from double-booking a seat even when the officer tried to do it unintentionally. The prototype would then be of great help for it would check whether a seat has already been made available to a different person before it has been booked to someone else.

With the help of the bar chart plotting the pre and post-test probabilities, it comes out clearly that the post-test probability of double-booking has been eradicated hence the 0 probability on the bar chart. A comparison of the pre and post-test probabilities simply shows that a probability of double-booking existed before introduction of the action but was reduced to 0 upon use of the prototype.

These results therefore generally suggest that the prototype solved the double-booking problem. This is because of introducing automated distribution and concurrency using programming techniques in the prototype. The booking officers would therefore end up successfully concurrently booking a seat from different locations with the use of distributed system software.

With the help of the bar chart plotting the pre-test probabilities, it comes out clearly that a probability of double-booking exists yet it shouldn't hence the pre-test results therefore generally suggest existence of a double-booking problem which occurs unintentionally and that comes with disgusting consequences. This is because of the booking officers trying to concurrently book a seat from different locations without use of distributed system software.

4.2 Specifying learning phase

This phase entails reflection on the results of the evaluation phase and on the success or failure of the conceptual framework. It is also in this phase that we reflect on the problem situation we're interested in and which can be solved using a particular practical method, action research as a research method, and on the problem in which the organisation is interested in. In addition, recommendations and limitations have been detailed herein.

4.2.1 Reflection on the results of evaluation

In general, it has been found out that the action had a positive effect on the business entity where it was realised that the probability of double-booking a seat was gotten rid of after introduction of the prototype. This means that it is now possible to concurrently book a seat in a distributed environment using such a distributed system prototype hence improving on the quality of service rendered to customers.

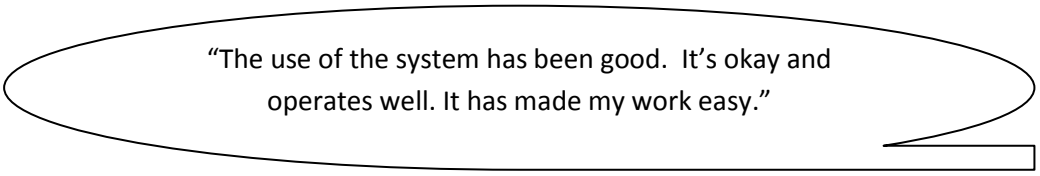
4.2.2 Reflection on success / failure of the conceptual framework

In order to achieve triangulation, we propose to use grounded theory analysis in order to reflect on the success / failure of the framework. Grounded theory entails three steps which are open coding, axial coding, and selective coding. Open coding involves identification of new variables in any stage of the Action Research cycle. Each variable refers to an attribute of a unit of analysis. (Kock, 2004) The variables identified earlier as attributes of the distributed system (unit of analysis) were system's efficiency, system's perception, system's usability, and system's effects. These are intervening variables that were identified in the initial stages of the

research and also relate with the independent and dependent variable. They act as a linkage between concurrency, distribution (independent variables) and service quality (dependent variables).

In order to confirm the success or failure of the conceptual framework that alludes concurrency and distribution have an effect on service quality, we hereby employ grounded theory coding to come up with the following concepts and categories that have been used to build the theoretical model shown in the following figure.

Open coding



The above quote and the following were associated with the **concepts** in italics.

- "The use of the system has been good. It's okay and operates well. It has made my work easy." - ***Difficulty***
- "Yes, I would recommend it for use elsewhere. This is because we have tested it and found it to have been good." - ***Satisfaction***
- "It is possible to confirm and takes 3 to 5 seconds." - ***Task time***
- "There is a high level of neatness and presentation as opposed to the previous manual system. The previous one had errors whilst the new one has improved on the order and presentation of our work." - ***Automation***
- "Yes, it's possible. A seat will only be available for booking to the first person." - ***Synchrony***
- "My service has been improved for a client finds progress in our use of technology." - ***Contribution***

- "It has been helpful for instance it has helped in booking and printing of customers details clearly on the ticket. It also alerts you in case of an error and has also helped prevent double-booking." **- *Experience***
- "It has played a good role for it has prevented double-booking in the sense that it prevents and alerts you when you try to book a seat that has already been booked." **- *Outcome***

Axial coding

The concepts above were theorized leading to the **categories** in the table below.

Concepts	Categories/Intervening variables
Difficulty	System's Usability
Satisfaction	
Task time	
Automation	System's efficiency
Synchrony	
Contribution	User's perception
Experience	
Outcome	System's effects

Table 1.5: Axial coding

Selective coding

The categories above were linked together leading to the theoretical model as shown in the figure below and confirmation of the conceptual framework in the figure that follows.

The model below generally implies that Service Quality was affected by the System's usability, system's efficiency, users' perception, and system's effects which in turn were deduced from the concepts.

Independent variables	Concepts	Categories/ Intervening variables	Dependent variables
Concurrency and distribution	Difficulty	System's usability	Service Quality
	Satisfaction		
	Task time		
	Automation	System's efficiency	
	Synchrzony		
	Contribution	Users' perception	
	Experience		
	Outcome	System's effects	

Table 1.6: Theoretical model

The theoretical model above therefore shows that Service Quality was affected by Concurrency and distribution. This is after a deduction from the qualitative information provided by the interviewees.

As mentioned earlier, we used the conceptual framework below, which represents a conceptualisation of the new business process state, to check on the kind of effect/contribution the variables had on one another.

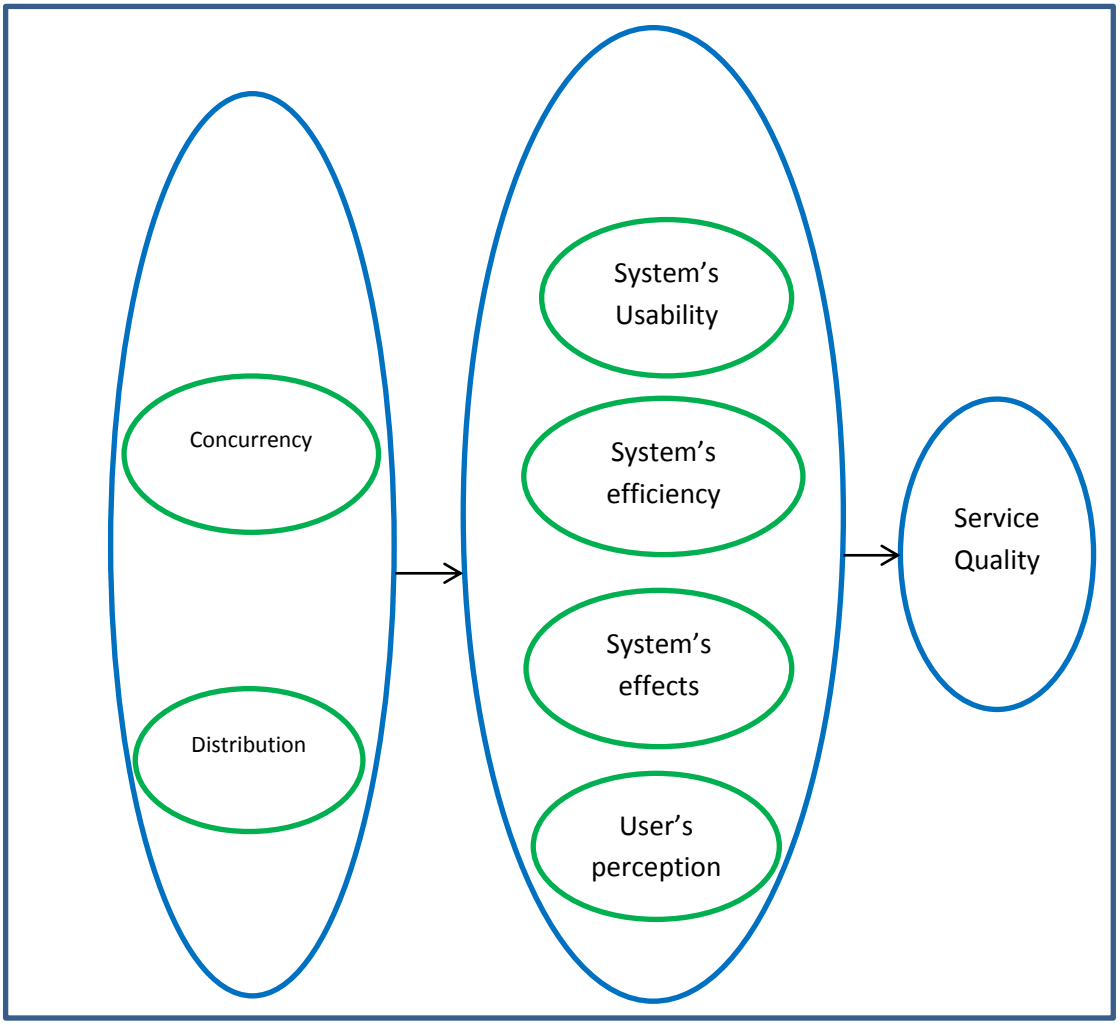


Fig 18: Conceptual framework

Following the above coding steps, we hereby confirm that the above framework, which also acted as our lens, has succeeded. We found out that introducing automated concurrency and distribution techniques through use of distributed systems had a positive effect on service quality based on the data collected and analysed. We therefore support, advocate for, and **recommend** it for adoption and further improvement made on it besides availing for critique.

4.2.3 Reflection on the problem and action research as a practical problem-solving method

It is hereby acknowledged that the double-booking problem was solved which existed due to lack of a distributed system that could automate concurrency and distribution. This acknowledgement is supported by the positive results that came out after introduction of the prototype and subsequent prototyping to ensure that the requirements were met.

We also support the use of action research method which helped us adopt a practical approach towards solving the concurrency problem in a distributed environment. It was of great importance and helped us achieve the research goal in not more than 1 iteration cycle. We therefore recommend its use as one of the best practical problem-solving methods.

Chapter Five

5.0 CONCLUSION AND RECOMMENDATIONS

5.1 Summary

The main objective of this research project was to determine the impact of automated concurrency and distribution on Service Quality with an aid of a distributed concurrent booking system prototype. Further, it explored the technologies that could be used to achieve such an objective where techniques such as use of Extended Mark-up Language (XML) in web services and locking algorithms were made use of. This exploration helped in development of the prototype whose effect on the concurrency problem was analysed in addition to the perception of the users towards it being evaluated.

The project was carried out in phases which included diagnosing, action planning, action taking, evaluating, and specifying learning. It's during the diagnosing phase that the problem was affirmed to be existent and the practitioners'/ officers' opinions, about the study being conducted with particular goals one of them being to help them in resolving their predicament, presented to the researcher. This was done and the idea welcomed with a subsequent phase of action planning which entailed prototyping and training.

Successively, the action was introduced as a prototype that was used during the action taking phase which was later followed by evaluation of the prototype and a reflection done on the evaluation's results as the last task. These tasks were done in collaboration with Imani Bus Company where the action research approach where both qualitative and quantitative methods were used to complement each other. Data collection was done using data archival and interviewing methods while analysis was mainly done using devised non-parametric statistical techniques.

5.2 Accomplishment of the Objectives

As mentioned above, it was seen that techniques such as use of web services based on the XML standard language, which are an implementation of the service-oriented architecture, and mutual locking algorithms were used well to the success of the project albeit the challenges encountered. Further, these techniques proved to be useful in that they could still be used even when a network hitch occurred.

As part of the system's efficiency, the users confirmed that the concurrent distributed system prototype received, stored, used, and presented the inputs as expected. They were used in the right manner in addition to the system being effective providing desired, neat, and presentable outputs in form of good tickets and charts. Overall, it was confirmed that the users had a good perception of the system prototype and recommended it for use elsewhere.

It was further determined after investigation that there was a positive effect on the booking process and generally on the quality of service offered that emanated from the system's use. This was after leveraging on automated concurrency and distribution to avoid double-booking, enable easy and fast booking, and to enable easy and quick retrieval of records.

5.3 Key contributions

In line with the aims of the research project, the study contributes by providing and supporting a conceptual framework and component model that were used to investigate and show the impact of automated concurrency and distribution on Service Quality. A positive impact was seen upon operationalization of the study. Secondly, it provides and encourages the use of a technique, which includes the use of a designed system architecture, which was crafted and followed to come up a concurrent distributed system prototype. Thirdly, it demonstrates and supports the use of the action research method in resolution of practical real-world problems. It was a good method that played a very important role in problem-solving.

It also demonstrates that due to the demerits of centralisation and asynchrony, which have involved the use of centralised servers in some current systems in distributed environments, it is important that we leverage on automated distribution which involves use of distributed fragments and replicas and automated concurrency which involves use of distributed algorithms, in order to improve on the quality of our services.

Conclusively, this research has successfully exhibited the importance of distributed systems' technologies which we can tap on to improve on the quality of services we provide in our day-to-day lives. It has been found out, through comparison of the service quality before and after introduction of distributed technologies, that automated concurrency and distribution have a **positive effect** on service quality.

5.4 Limitations and challenges encountered during the study

We hereby note that there were several limitations whose solutions could have bettered knowledge and discovery. Firstly, It was not possible to have 2 separate control groups during action taking where one could have acted as an experimental group as the other acted as a control group hence the quasi-experimental data collection method was used using 1 group within which simulation of a remote officer booking using a different terminal with the distributed system was done. It was also not possible to have the distributed system prototype tested by more than one organisation due to unavailability of enough resources and time-constraints hence external randomisation wasn't achieved.

However, internal randomisation within the organisation was met where different booking officers would use the system. Thirdly, coding was achieved through the response of 5 out of 8 booking officers within the branch who used the system and who were readily available for an interview. A larger sample size could have helped in realisation of new concepts. However, they gave a clear positive testament towards supporting the prototype.

Fourthly, there were no parametric tests done which could have led to a stronger motion and hypothesis. This is because the data collected did not represent a normal distribution even though the system was used as expected. The distribution had a significant level of negative Skewness and positive kurtosis hence could not be used to do T and Z-score tests. Some of the non-parametric tests such as the Whitney's and Signs tests could also not been done because the data collected showed a decrease in probability only. However, we still used a devised non-parametric test that was used to accurately analyse and present the probability change before and after the action.

5.5 Recommendations and Further research

This research work has therefore contributed towards solving the concurrency problem by providing and recommending a practical solution that was successfully tested besides demonstration of a framework and prototype that can be adopted in future to solve a similar problem. We recommend adoption of automated concurrency and distribution techniques in our systems so as to better our services and avoid some unpleasing incidences which hinder our holistic growth as a country. We also recognise some further research areas we came across in the course of the research project them being, Research work on distributed systems' ability to automatically update the fragments upon resumption of the network connection between the distributed entities and middleware. This can also be referred to as the ability to being autonomic.

This was done manually where the system would be prompted and one would be prevented from booking a seat in a remote fragment if the seat had already been booked before there happened to be a network halt. Additionally, further Research work can be done on discovery of reliable and affordable network linkage that would support the achievement and success of distributed systems in a real-time environment e.g. in nuclear power generation firms and hospitals. This will help in stabilisation of distributed systems as opposed to experiencing network halts which sometimes have an adverse effect on service provision.

6.0 REFERENCES

1. ALONSO, G., CASATI, F., KUNO, H. AND MACHIRAJU, V. (2004) Web Services. In: ALONSO, G., CASATI, F., KUNO, H. AND MACHIRAJU, V. (eds.) *Web Services – Concepts, Architectures and Applications.* Springer Verlag. pp. 123-149 [Online] Available from: http://ahvaz.ist.unomaha.edu/azad/temp/softarch/04-alonso-webservices-server-architecture-soa.pdf [Accessed 05th July 2014].
2. AVISON, D., LAU, F., MYERS, M., and NIELSEN, P. (1999), Action Research. *Association for Computing Machinery. Communications of the ACM.* [Online] 42 (1) pp.94-97 Available from: http://www.ic.unicamp.br/~wainer/cursos/2s2006/epistemico/act-acm.pdf [Accessed: 11th June 2015].
3. BASKERVILLE, R. (1999) Investigating Information Systems with Action Research. *Communications of the Association for Information Systems.* [Online] 2 Article 19 Available from: http://wise.vub.ac.be/thesis_info/action_research.pdf [Accessed: 11th June 2015].
4. Casavant, T.L. (eds). (1994) *Readings in Distributed Computing System*. 1st edition. United States of America: IEEE Computer Society Press.
5. EASTERBROOK, S., SINGER, J., STOREY, M., and DAMIAN, D. (2008) Selecting Empirical Methods for Software Engineering Research. In: SHULL, F., SINGER, J., and SJøBERG, D. (2008) (eds.) *Guide to Advanced Empirical Software Engineering.* [Online] London, Springer-Verlag London Limited pp. 285 – 311 Available from: http://read.pudn.com/downloads159/ebook/712887/2008-Guide%20to%20Advanced%20Empirical%20Software%20Engineering.pdf [Accessed: 10th June 2015].
6. F.M GHALEB, and REDA N.M (2012) Open-Gate: An Efficient Middleware System for Heterogeneous Distributed Databases. *International Journal of Computer Applications.* [Online] 45 (2) pp.44-49 Available from: http://research.ijcaonline.org/volume45/number2/pxc3879009.pdf[Accessed 05th July 2014].
7. GASHTI, M.Z., (2012) Investigating SOAP and XML Technologies in web service. *International Journal on soft computing.* [Online] 3 (4) Available

from: http://airccse.org/journal/ijsc/papers/3412ijsc02.pdf [Accessed 05th July 2014].

8. GULLEDGE, T. (2006) What is Integration? *Industrial Management and Data Systems.* [Online] 106 (1) pp.5-20 Available from: http://www.avyg86.dsl.pipex.com/ecom/0291060101.pdf [Accessed 07th July 2014].

9. GUPTA, S., SAROHA, K. and BHAWNA (2011) Fundamental Research of Distributed Database. *International Journal of Computer Science and Management Studies* [Online] 11 (2). p.138-146 Available from: http://www.ijcsms.com/journals/Volume%2011,%20Issue%2002,%20Aug%202011_Paper24.pdf [Accessed 04th July 2014].

10. HASHEMI, S. and HASHEMI S.Y. (2012) A Novel Service Oriented Architecture for integration of Information Systems in electronic city. *International journal of scientific and Technology Research.* [Online] 1 (11) pp.6-9 Available from: http://www.ijstr.org/final-print/dec2012/A-Novel-Service-Oriented-Architecture-For-Integration-Of-Information-Systems-In-Electronic-City.pdf[Accessed 07th July 2014].
11. JITENDRA, S. and Gupta, V.K. (2012) Concurrency Issues of Distributed Advance Transaction Process. *Research Journal of Recent Sciences* [Online] 1 (ISC-2011), pp.426-429. Available from: http://www.isca.in/rjrs/archive/iscsi/75.ISCA-ISC-2011-5CITS-20.pdf [Accessed: 04th July 2014].
12. KAUR, H. and KAUR, M. (2013) Concurrency Control in Distributed Database System. *International Journal of Advanced Research in Computer Science and Software Engineering* [Online] 1 (7), pp.1443-1447. Available from: http://www.ijarcsse.com/docs/papers/Volume_3/7_July2013/V3I7-0526.pdf [Accessed: 04th July 2014].
13. KESTER, Q. (2013) Using SOA with Web Services for effective Integration of Hospital Information Systems via an Enterprise Service Bus. *International Journal of Research in Engineering & Advanced Technology*. [Online] 1 (2) Available from: http://arxiv.org/ftp/arxiv/papers/1307/1307.7790.pdf [Accessed 07th July 2014].

14. KHAN, S. and HUSSAIN, W. (2008) *Component Based Software Development with EJB and .Net.* Malardalen University, Department of computer science and electronics Vasteras-Sweden Viewed 25th February 2014 http://www.idt.mdh.se/kurser/ct3340/archives/ht08/papersRM08/37.pdf

15. KOCK, N. (2004) The Three threats of action research: a discussion of methodological antidotes in the context of an information systems study. *Decision Support Systems* [Online] 37 (2004), pp.265-286. Available from: http://cits.tamiu.edu/kock/pubs/journals/2004JournalDSS/Kock2004.pdf [Accessed: 11th June 2015].

16. LESHEM, S. and TRAFFORD, V. (2007) Overlooking the conceptual framework. *Innovations in Education and Teaching International* [Online] 44 (1), pp.93-105. Available from: doi:10.1080/14703290601081407 [Accessed: 26th June 2015].

17. McKAY, J. and MARSHALL, P. (2001) The Dual imperatives of action research. *Information Technology & People.* [Online] 14 (1) pp.46-59 Available from: doi:10.1108/09593840110384771 [Accessed: 10th June 2015].

18. MOREIRA, R., ROCHA, A. and VASCONCELOS, J.B. (2005) *Middleware: The layer in between.* Viewed 25th February 2014 http://bdigital.ufp.pt/bitstream/10284/210/1/artigo9.pdf

19. MUMBAIQAR, S., and PADIYA, P. (2013) Web Services based on SOAP and Rest Principles. *International Journal of Scientific and Research Publications.* [Online] 3 (5) pp.1-4 Available from: http://www.ijsrp.org/research-paper-0513/ijsrp-p17115.pdf [Accessed 04th July 2014].

20. QUASIM, M.T, (2013) An Efficient approach for concurrency control in distributed database system. *Indian Streams Research Journal.* [Online] 3 (9) Available from: doi:10.9780/22307850 [Accessed 04th July 2014].

21. SCHMIDT D.C, (2002) Adaptive and Reflective Middleware for Distributed Real-time and Embedded Systems. In: VINCENTALLI, S.A.and SIFAKIS, J. (eds.) *Embedded Software.* Springer Verlag. pp. 282-293 [Online] Available from: doi:10.1007/3-540-45828-x_21 [Accessed 05th July 2014].

22. SOMMER, R., and SOMMER, B. (2002) Action Research. In: Sommer, R. & Sommer, B. (eds.) *A Practical Guide to Behavioral Research*. 5th edition. Oxford University Press. pp. 211-217 [Online] Available from: http://psychology.ucdavis.edu/sommerb/sommerdemo/doing/actionResearch.pdf [Accessed: 22nd August 2013].

23. SURUGIU, I. (2012) Integration of Information Technologies in Enterprise Application Development. *Database Systems Journal.* [Online] 3 (1) pp.21-31 Available from: http://www.dbjournal.ro/archive/7/7_3.pdf [Accessed 07th July 2014].

24. TANENBAUM, A.S. (eds). (2007) *Distributed Systems: Principles and paradigms*. 2nd edition. United States of America: Pearson Prentice Hall.

7.0 BUDGET

Estimated costs are as shown below.

a. 2 Computers Ksh.60000
b. Safaricom WiMax Connection Ksh.20000 per month
c. Printers Ksh.11000
d. Routers Ksh.6000
e. Telephone charges Ksh.1400
f. Printing Costs Ksh.1000
g. Binding Ksh.100
h. Photocopying Costs Ksh.500
i. Training Costs Ksh.5000
j. Travelling Ksh.6000
k. Miscellaneous Ksh.10000

8.0 APPENDICES

APPENDIX I: PRE-TEST INTERVIEW

1. Tell me about double-booking…. What is it?
2. Do you know the main reason as to why it happens and if so, please explain..
3. What are your opinions about it? How do you feel about it?
4. How frequently does it happen and how do you deal with it once it happens? Are you usually in a position to somehow remedy the situation?
5. Please describe the image that's usually portrayed once it happens?
6. Does the double-booking scenario happen intentionally? Please explain…
7. Please describe of any lesson learnt from such a case if any….
8. Please explain about any measure in place to solve such a problem if any and explain of how effective it has been…
9. How do the customers react upon the occurrence of a double-booked seat?
10. Does double-booking affect your service to the customers?
11. On average, how many seats do you **successfully** book per week?
12. What's the maximum number of seats you've ever double-booked per day since you began working for this company?
13. What's the least number of seats you've ever double-booked per day since you began working for this company?
14. On average, how many customers complain of double-booking per week if any?
15. What do you think may happen in 5 years to come if double-booking persists and is not solved at all?
16. What do you think needs to be done in order to help you be the best in the transport industry?

17. Who bears the burden of a seat that has been double-booked, what kind of a burden is it and to what extent is it borne?
18. How easy / hard is it to deal with this situation? Please explain …. Have you tried to avoid it before? What to do you do to try avoiding it and does it still occur even when trying to avoid it?
19. Do you know of other methods that can be used to solve this problem?
20. How soon would you want this problem solved?

APPENDIX II: POST-TEST INTERVIEW

1. Please describe your interaction with the system…
2. Would you recommend such a system to any other organisation and why?
3. Do you experience a terminal / computer / system halt when using the system? If so, please explain ….
4. How possible is it for a customer to confirm a booking and how long does it take for you to confirm?
5. How easy / hard is it for you to make mistakes when using the system?
6. What level of neatness and presentation is present if any and was it there before introduction of the system?
7. Please describe of any lesson learnt from such a case if any….
8. How possible is it now for you to concurrently book a seat without any case of double-booking?
9. Please describe how the system handles every raw data that's fed into the system. What's the level of accuracy in the system?
10. Please explain how the system has affected your service to the customers…
11. What's the maximum number of seats you've ever double-booked per day since you began using the system?
12. What's the least number of seats you've ever double-booked per day since you began using the system?
13. On average, how many customers would complain of double-booking per week if any?
14. Generally, how have you found the system? Has it helped you?
15. What opinions do you have of the system? What role has it played in the problem-solving process?
16. How far do you think you will be as an individual and as a company in 2 years' time as a result of using the system?
17. What are the advantages of using the system if any?
18. What are the disadvantages of using the system if any?

19. In general, what's your opinion about the quality of service offered through the use of the system?

APPENDIX III: RESEARCH REQUEST COVER LETTER

SHADRACK .M. NGUMBAU,

P.O. BOX 92285 - 80102,

CHANGAMWE, KENYA.

✆: +254712999267

✉: shadmwanzia@gmail.com

27th October, 2015.

TO WHOM IT MAY CONCERN,

IMANI DVD COACH LIMITED,

Dear Sir/Madam:

RE: DISTRIBUTED CONCURRENT BOOKING SYSTEM

This is to request for an opportunity to collaborate with you on my research project that entails investigating the **impact** of automated distribution and concurrency **on** Service Quality through the aid of a distributed and concurrent bus seat booking system developed for Imani Coach Limited. I am a Systems Integration Engineer and a Master's degree course student at the University of Nairobi.

Please find the attached files i.e. CV and Detailed Document. If possible, please reply via my email address / mobile number in the contacts above on or before 28/10/2015.

I hope you shall consider my request.

Yours Faithfully,

Shadrack Mwanzia Ngumbau.

APPENDIX IV: LOG-IN PAGE CODE

```
<!DOCTYPE html [<!ENTITY nbsp " ">]>
<html xmlns="http://www.w3.org/1999/xhtml"
xmlns:f="http://java.sun.com/jsf/core"
xmlns:h="http://java.sun.com/jsf/html"
xmlns:c="http://java.sun.com/jsp/jstl/core">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8"></meta>
<title>BOOKING PORTAL</title>
<link href="css/main_style.css" rel="stylesheet" type="text/css"></link>
<link href="css/main_style1.css" rel="stylesheet" type="text/css"></link>
<link href="css/style2.css" rel="stylesheet" type="text/css"></link>
<link href="css/global.css" rel="stylesheet" type="text/css"></link>
<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/scripts.js"></script>
<link href="utils/css/ticker.css" rel="stylesheet" type="text/css"></link>
<script src="utils/jscripts/jquery.js" type="text/javascript"></script>
<script src="utils/jscripts/jquery_ticker.js" type="text/javascript"></script>
<script src="utils/jscripts/ticker.js" type="text/javascript"></script>
</head>
<body style="background-color: #D8D8D8;">
<div style=" margin-top: 120px; margin-bottom: 250px; margin-right: 550px;
margin-left: 560px;
width: 350px; border-radius: 25px; background-color: #FFFFFF; height: 300px;
padding: 20px 20px 20px;" >
<h1 style="background-color: #FFFFFF; color: #FE9A2E; text-align: center; float:
center; width: 302px;
font-weight:bold; font-size: 35px;">IMANI COACH</h1>
<h:form style="background-color: #FFFFFF;" id ="loginform23" >
```

```
<h:outputLabel style="color: black; font-weight:
bold;">Username:</h:outputLabel> 

<h:inputText  style="  background-color: #F2F2F2; border-radius: 5px; height:
40px; width: 233px;

/*width: 291px;*/ margin-left: 0px;" id="username"
value="#{conn.mobilenumber}" /><br/><br/>

<h:outputLabel style="color: black; font-weight:
bold;">Password:</h:outputLabel> 

<h:inputSecret style="background-color: #F2F2F2; border-radius: 5px; height:
40px; width: 233px;

/*width: 291px;*/ margin-left: 0px;" id="password" value="#{conn.pin}"
/><br/><br/>

<h:commandButton  style=" border-radius: 5px; background-color: #FE9A2E; float:
center; height: 40px;

width: 348px; margin-right: 10px; font-weight: bold;" action="#{conn.connect()}"
value="Log in"/><br/><br/>

</h:form>

</div>

</body>

</html>
```

APPENDIX V: MIDDLEWARE WEB-SERVICE CODE

```
package com.dmw.dmww;

import com.sto.stosws.Stosws_Service;

import com.sto.stows.InterruptedException_Exception;

import com.sto.stows.Stows_Service;

import javax.jws.WebService;

import javax.jws.WebMethod;

import javax.jws.WebParam;

import javax.xml.ws.WebServiceRef;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.net.InetSocketAddress;

import java.net.Socket;

import java.net.URL;

import java.net.URLConnection;

import java.nio.charset.Charset;

import java.util.logging.Level;

import java.util.logging.Logger;

/**

 *

 * @author Shadrack

 */

@WebService(serviceName = "dmww")

public class dmww {

   @WebServiceRef(wsdlLocation = "WEB-
INF/wsdl/169.254.29.64_8080/stos/stosws.wsdl")

   private Stosws_Service service_1;

   @WebServiceRef(wsdlLocation = "WEB-INF/wsdl/localhost_8080/sto/stows.wsdl")
```

```
private Stows_Service service;
public static String callURL(String myURL) {
   System.out.println("Requested URL:" + myURL);
   StringBuilder sb = new StringBuilder();
   URLConnection urlConn = null;
   InputStreamReader in = null;
   try {
      URL url = new URL(myURL);
      urlConn = url.openConnection();
      if (urlConn != null) {
         urlConn.setReadTimeout(300 * 1000);
      }
      if (urlConn != null && urlConn.getInputStream() != null) {
         in = new InputStreamReader(urlConn.getInputStream(),
               Charset.defaultCharset());
         BufferedReader bufferedReader = new BufferedReader(in);
         if (bufferedReader != null) {
            int cp;
            while ((cp = bufferedReader.read()) != -1) {
               sb.append((char) cp);
            }
            bufferedReader.close();
         }
      }
      in.close();
   } catch (Exception e) {
      throw new RuntimeException("Exception while calling URL:" + myURL, e);
   }
   return sb.toString();
```

```
    }

    @WebMethod(operationName = "checkstatusdw")

    public synchronized String checkstatusdw(@WebParam(name = "station") String sta,

            @WebParam(name = "seatnumbercumber") String seatnumberco,
@WebParam(name = "departuredate")

            String departuredate, @WebParam(name = "departuretime") String departureti,

            @WebParam(name = "destination") String destin) throws InterruptedException {

        String seatnumberc = seatnumberco;

        String dateoftravelc = departuredate;

        String departuretimec = departureti;

        String destinationc = destin;

        String stationc = sta;

        String tring = null;

        String status = null;

        Buffer sharedLocationc = (Buffer) new SynchBuffer();

        try {

            Socket socket = new Socket();

            int timeout = 3000;

            socket.connect(new InetSocketAddress("localhost", 80), timeout);

            socket.setSoTimeout(timeout);

            String li = seatnumberc;

            URL url = new URL("http://localhost:8080/dmw/dmwcst?seatnumber=" +
seatnumberc +

                                    "&departuredate=" + dateoftravelc + "&departuretime=" +
departuretimec + "&destination="

                                    + destinationc + "&station=" + stationc);

            tring = callURL(url.toString());

            System.out.println("tringfr:::" + tring);

            status = tring;

            System.out.println("status::" + status);
```

```
    } catch (Exception e) {
        e.printStackTrace();
        System.out.println("exception....");
        System.out.println("status:::" + status);
        //status = null;
    }
    return status;
  }
  @WebMethod(operationName = "checkstatusdwsto")
  public synchronized String checkstatusdwsto(@WebParam(name =
"seatnumbercumber") String seatnumberco,
        @WebParam(name = "departuredate") String departuredate, @WebParam(name
= "departuretime")
        String departureti, @WebParam(name = "destination") String destin) throws
InterruptedException {
    String seatnumberc = seatnumberco;
    String dateoftravelc = departuredate;
    String departuretimec = departureti;
    String destinationc = destin;
    Buffer sharedLocationc = (Buffer) new SynchBuffer();
    String status = null;
    try {
        status = checkstatus(seatnumberc, dateoftravelc, departuretimec, destinationc);
        System.out.println("status::" + status);
    } catch (Exception e) {
        e.printStackTrace();
                }
    return status;
  }
  @WebMethod(operationName = "checkstatusdwstos")
```

```
   public synchronized String checkstatusdwstos(@WebParam(name =
"seatnumbercumber") String seatnumberco,

           @WebParam(name = "departuredate") String departuredate, @WebParam(name
= "departuretime")

           String departureti, @WebParam(name = "destination") String destin) throws
InterruptedException {

       String seatnumberc = seatnumberco;

       String dateoftravelc = departuredate;

       String departuretimec = departureti;

       String destinationc = destin;

       Buffer sharedLocationc = (Buffer) new SynchBuffer();

       String status = null;

       try {

          status = checkstatusws(seatnumberc, dateoftravelc, departuretimec, destinationc);


          System.out.println("status::" + status);

       } catch (Exception e) {

          e.printStackTrace();

       }

       return status;

   }

   /** This is a sample web service operation */

   @WebMethod(operationName = "processssdw")

   public synchronized String processssdw(@WebParam(name = "seatnumbercumber")
String seatnumbercumber,

           @WebParam(name = "passengersid") String passengersid, @WebParam(name =
"passengersfirstname")

           String passengersfirstname, @WebParam(name = "passengersmiddlename") String
passengersmiddlename,

           @WebParam(name = "passengerslastname") String passengerslastname,
@WebParam(name = "dateoftravel")

           String dateoftravel, @WebParam(name = "reportingtime") String reportingtime,
@WebParam(name = "departuretime")
```

```
        String departuretime, @WebParam(name = "fare") String fare, @WebParam(name
= "destination") String destination,

        @WebParam(name = "origin") String origin) throws InterruptedException {

    String seatnp = seatnumbercumber;

    String pasid = passengersid;

    String passfname = passengersfirstname;

    String passmname = passengersmiddlename;

    String passlname = passengerslastname;

    String daoftr = dateoftravel;

    String reptime = reportingtime;

    String depti = departuretime;

    String far = fare;

    String desti = destination;

    String ori = origin;

    String responsedw = null;

    if ("A23".equals(seatnp) || "A24".equals(seatnp) || "A25".equals(seatnp) ||
"A26".equals(seatnp)

                || "A27".equals(seatnp) || "A28".equals(seatnp) || "A15".equals(seatnp)
|| "A16".equals(seatnp)

                || "A17".equals(seatnp) || "A18".equals(seatnp) || "A19".equals(seatnp)
|| "A20".equals(seatnp)

                || "A21".equals(seatnp) || "A22".equals(seatnp)) {

        try {

            responsedw = processws(seatnp, pasid, passfname, passmname, passlname,
daoftr, reptime, depti, far, desti, ori);

        } catch (com.sto.stosws.InterruptedException_Exception ex) {

            Logger.getLogger(dmww.class.getName()).log(Level.SEVERE, null, ex);

        }

            } else {

        try {

            responsedw = process(seatnp, pasid, passfname, passmname, passlname, daoftr,
reptime, depti, far, desti, ori);
```

```
            } catch (InterruptedException_Exception ex) {
                Logger.getLogger(dmww.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
        return responsedw;
    }

    private String checkstatus(java.lang.String seatnumber, java.lang.String departuredate,
java.lang.String departuretime, java.lang.String destination) throws
InterruptedException_Exception {
        com.sto.stows.Stows port = service.getStowsPort();
        return port.checkstatus(seatnumber, departuredate, departuretime, destination);
    }

    private String process(java.lang.String seatnumber, java.lang.String passengersid,
java.lang.String passengersfirstname, java.lang.String passengersmiddlename,
java.lang.String passengerslastname, java.lang.String dateoftravel, java.lang.String
reportingtime, java.lang.String departuretime, java.lang.String fare, java.lang.String
destination, java.lang.String origin) throws InterruptedException_Exception {
        com.sto.stows.Stows port = service.getStowsPort();
        return port.process(seatnumber, passengersid, passengersfirstname,
passengersmiddlename, passengerslastname, dateoftravel, reportingtime, departuretime,
fare, destination, origin);
    }

    private String processws(java.lang.String seatnumber, java.lang.String passengersid,
java.lang.String passengersfirstname, java.lang.String passengersmiddlename,
java.lang.String passengerslastname, java.lang.String dateoftravel, java.lang.String
reportingtime, java.lang.String departuretime, java.lang.String fare, java.lang.String
destination, java.lang.String origin) throws
com.sto.stosws.InterruptedException_Exception {
        com.sto.stosws.Stosws port = service_1.getStoswsPort();
        return port.processws(seatnumber, passengersid, passengersfirstname,
passengersmiddlename, passengerslastname, dateoftravel, reportingtime, departuretime,
fare, destination, origin);
    }

    private String checkstatusws(java.lang.String seatnumber, java.lang.String
departuredate, java.lang.String departuretime, java.lang.String destination) throws
com.sto.stosws.InterruptedException_Exception {
```

```
        com.sto.stosws.Stosws port = service_1.getStoswsPort();
        return port.checkstatusws(seatnumber, departuredate, departuretime, destination);
    }
}
```

APPENDIX VI: PARTICIPATORY OBSERVATION (FIELD NOTES)

Date: 29/12/2015

Location: Imani DVD Coach Booking Office : Erusha House

Description:

There is a booking officer busy serving clients interested in travelling to Mombasa. It is a busy office that entails wooing of customers as they pass by. In the process a SACCO chairman pops in to request of what transpired in the latest double-booking case. The booking officer narrates of how his colleague assured him of 10 seats available in a bus from Embu destined for Mombasa.

The bus is to pass by Nairobi, pick passengers and head to Nairobi. The booking officer books the seats and issues tickets to new customers interested in travelling to Mombasa. The bus turns out to be full upon arrival in Nairobi leading to the annoyance of the booking officer as 1 client misses a seat. However, the booking officer narrates that he had anticipated double-booing from experience hence had reserved an extra seat. He then books this seat for the client.