

# Graduation Thesis

## Asian Restaurant Management System

<b>Name</b>	HOSSAIN SYED ABIR(阿贝尔)
<b>Major</b>	Computer Science and Technology
<b>Supervisor</b>	Prof. Liang Zhao

SHENYANG AEROSPACE UNIVERSITY

December 14, 2018

# Asian Restaurant Management System

College	Computer
Major	Computer Science and Technology
Class	140103
Stuent No.	14031274
Name	text
Supervisor	HOSSAIN SYED ABIR(阿贝尔)

SHENYANG AEROSPACE UNIVERSITY

December 14, 2018

# Declaration

To the best of my knowledge and belief, I certify that this thesis does not:

- - Incorporate any material previously submitted for a degree or diploma in any institution of higher education without acknowledgement.
  - Contain any material previously published or written by other person except where due reference is made in the text.
  - Contain any defamatory material.

I also grant permission to Shenyang Aerospace University to make duplicate copies of my thesis as required.

Signature.....

Date.....

# ABSTRACT

This report documents the process of designing, developing and testing a software system to be used in a restaurant; usually given the name restaurant management system. The restaurant management system is there to help communication between all teams within a restaurant by minimizing the probability of human errors.

We propose to build a software project that can efficiently handle and manage various activities of a restaurant and all these activities will be happening under the supervision of the administrator. The businesses in restaurants are now growing constantly. At the same time, the need for managing its operations and tasks arises. The best way to optimize these activities is growing the business online as well. Today's generation encourages high-tech services especially over the Internet. Hence the project is developed proficiently to help restaurant owners automate their business operations. This project serves the best way of maintaining customer's information and caters their needs.

## Contents

<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
1.1	EXISTING SYSTEM . . . . .	1
1.1.1	DISADVANTAGES OF EXISTING SYSTEM . . . . .	1
1.2	PROPOSED SYSTEM . . . . .	2
1.2.1	ADVANTAGES OF PROPOSED SYSTEM . . . . .	2
1.3	MODULES . . . . .	2
1.3.1	MODULES DESCRIPTION . . . . .	2
1.4	Summary of Chapters . . . . .	3
1.5	Closing Remarks . . . . .	3
<b>2</b>	<b>Chapter Two</b>	<b>4</b>
2.1	Chapter Overview . . . . .	4
2.2	Point-of-Sale (POS) Systems . . . . .	4
2.3	Platform Choice . . . . .	4
2.4	UML . . . . .	4
2.5	Software Choice . . . . .	5
2.6	Requirement Gathering . . . . .	5
2.7	Development Methodology . . . . .	7
2.7.1	. . . . .	7
2.8	Chapter Summary . . . . .	7
<b>3</b>	<b>Chapter Three</b>	<b>8</b>
3.1	Chapter Overview . . . . .	8
3.2	Stakeholder Identification . . . . .	8
3.3	Use Cases . . . . .	9
3.4	Features . . . . .	10
3.5	Measureable Goals and Requirements . . . . .	10
3.6	Chapter Summary . . . . .	10

<b>4</b>	<b>Chapter Four</b>	<b>11</b>
4.1	Chapter Overview . . . . .	11
4.2	Introduction . . . . .	11
4.3	Component Diagram . . . . .	11
4.4	Data Storage . . . . .	11
4.4.1	Relational Database Management System (RDBMS) . . . . .	12
4.4.2	Extensible Markup Language (XML) . . . . .	12
4.4.3	Storage Method Chosen . . . . .	12
4.4.4	Normalisation . . . . .	13
4.5	Chapter Summary . . . . .	14
<b>5</b>	<b>Chapter Five</b>	<b>15</b>
5.1	Chapter Overview . . . . .	15
5.2	Chapter Summary . . . . .	15
<b>6</b>	<b>Chapter Sixth</b>	<b>16</b>
6.1	Chapter Overview . . . . .	16
6.2	Chapter Summary . . . . .	16

## List of Figures

1	UML diagram for Asian Restaurant Management system. . . . .	5
2	Boehm's cost model [1] . . . . .	6
3	Use case diagram showing some of the major fearures with in the asian restaurant management system . . . . .	9
4	Component diagram showing the higher level architecture of the system.	11
5	Database comparison diagram [2] . . . . .	13

## List of Tables

1	A table showing the stakeholders of the project . . . . .	8
2	A table showing the proposed system features and allocated priorities. . .	10

# 1 INTRODUCTION

The concept of restaurant table order management system, since it is java application, I will keep everything as simple as possible. The project consists in an java application that can be used by employees in a restaurant to handle the clients, their orders and can help them easily find free tables or place orders. This application, created mainly for proof of proper user-java interaction. The restaurant menu is organized by categories (appetizers, soups,fig salads, entrees, sides and drinks) of menu items. Each menu item has a chef, preparation instructions and associated ingredients. The ingredients are identified by their ingredient id and the quality of the ingredient needed to prepare a particular recipe, the unit of measure and a name.

"Restaurant Management System(RMS)" is java application to restaurant management. This system wake to provide service facility to rastaurant and also to the customer. The services that are provided is food ordering and home delivery by the customer through the system, customer information management and waiter information management, menu information management and report. Main objective build the system, ordering, and home delivery management will become easier and systematic to replace traditional system.

## 1.1 EXISTING SYSTEM

- Restaurant services such as making processing orders, and delivering meals generally require waiters to input customer information and then transmit the orders to kitchen for meal preparation. When the customer pays the bill, the amount due is calculated by the cashier. Although this procedure is simple, it may significantly increase the workload of waiters and even cause errors in meal ordering or in prioritizing customers, especially when the number of customers suddenly increases during busy hours, which can seriously degrade the overall service quality.
- A very commonly implemented system, currently being used by numerous restaurants and chains all over the world, is the electronic point-sale terminal system.
- Here the waiters generally take the order from the customer and head onto a terminal, where they can feed the order into a computer . The order can the be trasnmitted to to the kitchen automatically via the terminal through the a network, or it may event be delivered manually by the server to the kitchen.

### 1.1.1 DISADVANTAGES OF EXISTING SYSTEM

- Although a huge improvement over the pen and paper still prevalent over world, this does not have much value addition for the customer and mostly only benefits the establishment and the administration of the establishment.
- It may significantly increase the workload of waiters and even cause errors in meal ordering or in prioritizing customers, especially when the number of customers suddenly increases during busy hours, which can seriously grade the overall service quality.

## 1.2 PROPOSED SYSTEM

- The system will consist of the following main components: The backend, which is made up of the web server and the database, and the frontends that include both the patron frontend and the administration or the kitchen frontend
- this system is based on the very popular model-View-Controller (MVC) architecture. MVC is most commonly used in websites, very popular and tried and tested. None of the frontends directly "talk" to the database. They instead rely on RESTful services that can be used to perform CRUD operations on the database.

### 1.2.1 ADVANTAGES OF PROPOSED SYSTEM

- The most important components of this system are the database and the frontends
- Providing value to both the business and its patron is an important objective, but we believe that one follows the other.
- Following that belief, the customer is given a whole lot of importance.

## 1.3 MODULES

- Customer
- Administrator
- Customer Ordering and home delivery
- Feedback Module
- Menu Module
- Generate Report Module

### 1.3.1 MODULES DESCRIPTION

- **Customer**  
This project module consists in a Java application that can be used by employees in a restaurant to handle the clients, their orders and can help them easily find free tables or place orders. This application, created mainly for proof of proper user interaction.
- **Administrator**  
Administrator is the person who will manage the entire system. This type of user will also do maintenance and control the application of this system. Administrator takes a responsibility to add a new customer, new menu into database, and etc.
- **Customer Ordering**  
Customer ordering and reservation module provides a form that needs to be fully filled in terms of ordering food.



- **Menu Module**

Menu module is food that restaurant prepared for customer. This module, customer can view the menu and make decision for order.

- **Generate Report Module**

System provides an option for generate a report. The contents of the report as following:

1. The report of customer ordering.
2. Customers information
3. Comment or satisfaction score insert into feedback form.

## **1.4 Summary of Chapeters**

The rest of this report consists of the following chapters:

- Background: Background investigation into the problem.
- Requirement Analysis: Requirements of the system including stakeholder identification, list of features and tabulated requirements.
- Design: Project design process using several diagrammatic techniques.
- Implementation: Discusses the implementation of the software with the help of diagrams and pseudocode.
- Results: Illustrates the system using screenshots.
- Testing: Documents how the system was tested.
- Conclusion: Project conclusion with future development ideas.

## **1.5 Closing Remarks**

This chapter has introduced the foundations of the project. The next chapter gives some background investigation into the problem.

## 2 Chapter Two

### Background

#### 2.1 Chapter Overview

This chapter gives an insight to Point of Sale (POS) systems similar in nature to that of the one being developed in this project. It also gives a brief introduction to the importance of requirement gathering, a discussion on the development methodologies available as well as a justification on the platform and software used in this project.

#### 2.2 Point-of-Sale (POS) Systems

According to A. Nutt [12], POS systems first dated back to the 1870s, when James Ritty became frustrated with dishonest employees stealing money from the customers in a saloon in Dayton, Ohio. With the inspiration from a counter on a ship's propeller that counted the number of revolutions, he and his brother in 1879 developed the first cash register called 'Ritty's Incorruptible Cashier'. By the 1970s, the first computerised cash register was developed that was basically a mainframe but packaged as a store controller that had the ability to control the registers. Then in the 1980's, cash registers began to be PC operated which meant that many of the basic PC functions were now available. Today, the POS systems are much faster, safer and reliable and with the introduction of credit cards and direct communication to the credit card company, transactions are almost instant.

#### 2.3 Platform Choice

Choosing a suitable platform normally goes down to the programmers experience and the type of software to be developed. The restaurant management system could be developed as a web application or a standalone application but must also be widely supported and platform-independent. Therefore as the developer has minimal or no experience in web programming, the decision was taken to develop a standalone application. The next decision was to decide on a programming language, with the developer having previous experience in C, C++ and Java. This decision was fairly easy and Java was the selected programming language as the developer has great knowledge in the Java Database Connectivity (JDBC) API 1 that allows database-independent connectivity between the Java programming language and numerous databases.

#### 2.4 UML

Diagrammatic techniques are used to visualise, construct and document software systems under development. The most general modelling language to describe both the structure and behaviour of a software system is Unified Modelling language (UML) created by the Object management group. The diagrams one can create using UML can be shown by a class diagram (Figure 1). Throughout this report, numerous models defined within the UML class diagram (Figure 1) will be used to graphically represent the system.

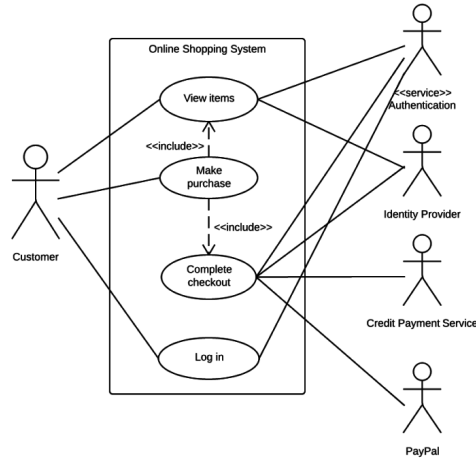


Figure 1: UML diagram for Asian Restaurant Management system.

## 2.5 Software Choice

In software development, the use of integrated development environments (IDE) can increase the efficiency of a programmer. An IDE is a software application that consists of a source code editor, compiler and debugging tools with its main aim to assist the programmer. Simple notepads are not strictly IDEs but can do the same job with the assistance of a compiler. The two most popular IDEs available for Java programming are Netbeans and Eclipse as they are free, support multiple platforms and offer many features including integrated version control and debugging tools. The two main problems with IDE's are that due to the wealth of features available and plug-in support, there is an associated cost, as their performance is poor and in particular they require more memory and processing power than a standard text editor. For this project Netbeans was the chosen IDE, as the developer has more experience and knowledge of the Netbeans graphical user interface.

## 2.6 Requirement Gathering

Requirement gathering is a very important step in software engineering. According to an article written by Craig Murphy [1], Boehm's cost model discusses how discovering errors at an early stage of software development can reduce overall costs. Figure 2 is an example of Boehm's cost model and uses the waterfall technique to give a visual insight into how important gathering accurate requirements can be when the project has a strict deadline to work to. Roughly a £1 cost of change in the requirement gathering can cost up to £100 in the testing stage and £1000 in the deployment stage to fix.

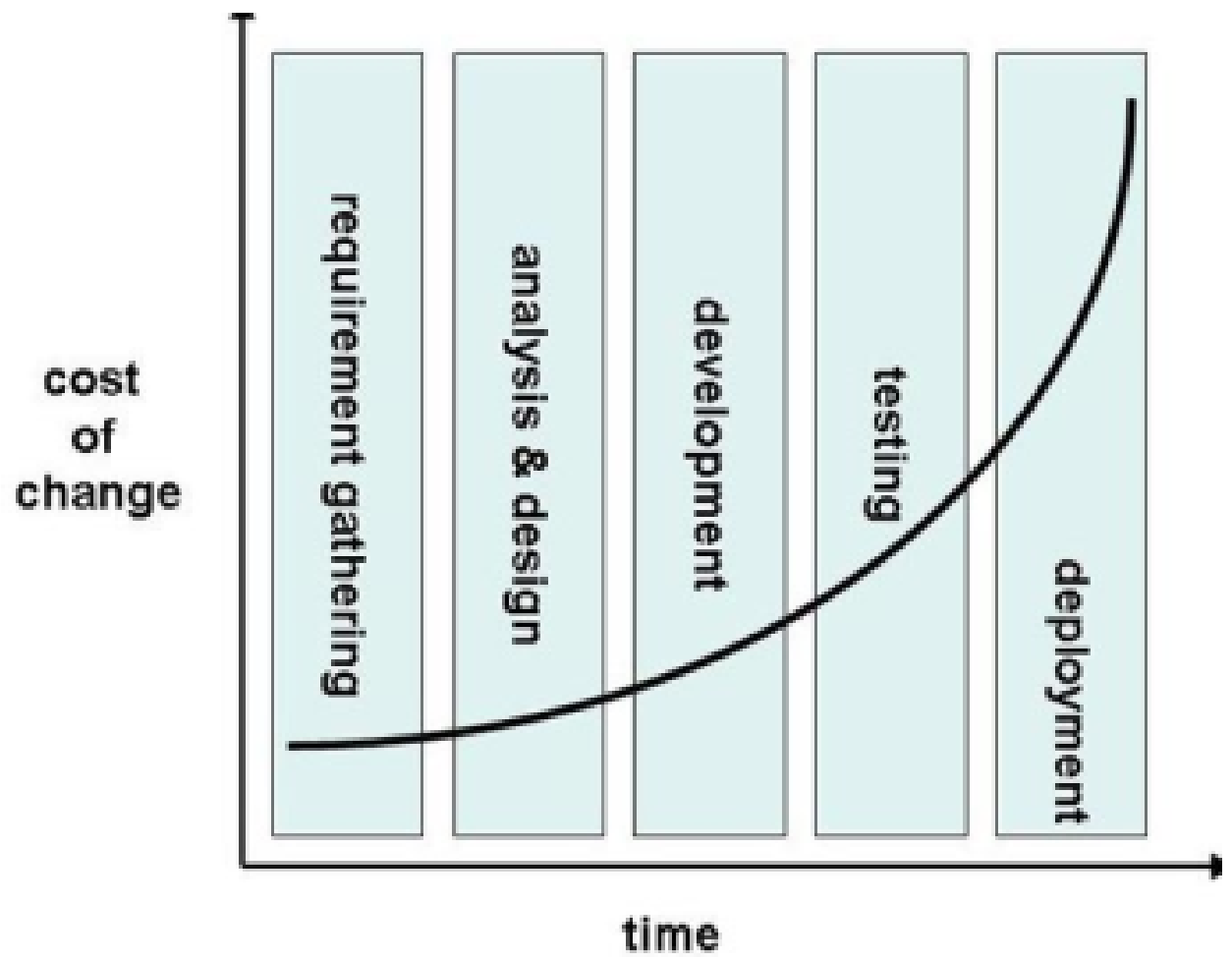


Figure 2: Boehm's cost model [1]

## 2.7 Development Methodology

A software development methodology is a framework used to plan the design of a software system controlling the process of development. According to Geoffrey Elliott [3], software development methodologies first emerged in the 1960's with systems development life cycles (SDLC) being considered the first formalised methodology. Since then there have been numerous popular software development approaches including the waterfall model, prototyping, incremental, spiral and agile. The agile methodology refers to a collection of different agile methods.

This project will be based on Extreme programming (XP) which is one of these agile methodologies using an iterative based framework. Each iteration has a development cycle very similar to the waterfall model consisting of planning, requirements analysis, design, development and testing. At each iteration, the business representative sometimes referred to as the 'customer' is given a demonstration to promote useful feedback. This type of methodology reduces risk and lets the project adapt to requested changes quickly with minimum cost.

According to the Agile Manifesto [4], some of the main principles behind the agile methodology are:

### 2.7.1

- Customer satisfaction by rapid, continuous delivery of useful software.
- Working software that is delivered frequently.
- A welcomed late change in requirements.
- Simplicity.
- Regular adaptation to changing circumstances.

## 2.8 Chapter Summary

This chapter has given examples of POS systems that are directly related to this project as well as general information about requirement gathering and development methodologies. The next chapter starts the process of the development methodology by generating the requirements of the system.

# 3 Chapter Three

## Requirement Analysis

### 3.1 Chapter Overview

This chapter will look at the stakeholders of the system and then discuss the required features using a use case diagram to illustrate.

### 3.2 Stakeholder Identification

As defined in the Business Dictionary [5], a stakeholder could be a person, group, organisation that has direct or indirect stake in an organization because it can affect or be affected by the organization's actions, objectives and policies.

Hence, stakeholders can be split into two groups; internal and external with each stakeholder contributing directly or indirectly towards the business activities.

As an example, any system where there exists customer communication, that customer will be a non-financial beneficiary stakeholder.

According to an article written by Ian Alexander [6], the person, group, organisation, or system is a stakeholder if they can be defined by any one of the following four questions:

Who needs to be consulted on the scope of this project?

Who has an input to the budget of this project?

Who can support or harm this project politically?

Who can provide guidance on the usability, functionality, and required qualities (reliability, safety, lifetime, maintainability, . . . ) of the system under development?

Therefore using these 4 questions as a guide, we can generate a list (1) of the stakeholders in this project.

Table 1: A table showing the stakeholders of the project

Stakeholder	Role
Carl Abernethy	Project developer.
Prof. Chris Taylor	Project supervisor.
Management	User of the management application.
Waiters	User of the restaurant application.
Kitchen Staff	User of the kitchen application.
Bar Staff	User of the restaurant application.
Customer	Indirect user of the system.
Cashier	External stakeholder; accepts payment.

### 3.3 Use Cases

A use case diagram that is part of the Unified Modelling Language (UML) which was introduced in Section 2.4 gives a graphical overview of the functionality of a system. A use case diagram consists of actors that are normally the stakeholders of the system and their use cases commonly defined as goals. Figure 3 shows several use case scenarios of the system that convey how the stakeholders interact with the features to achieve a business requirement.

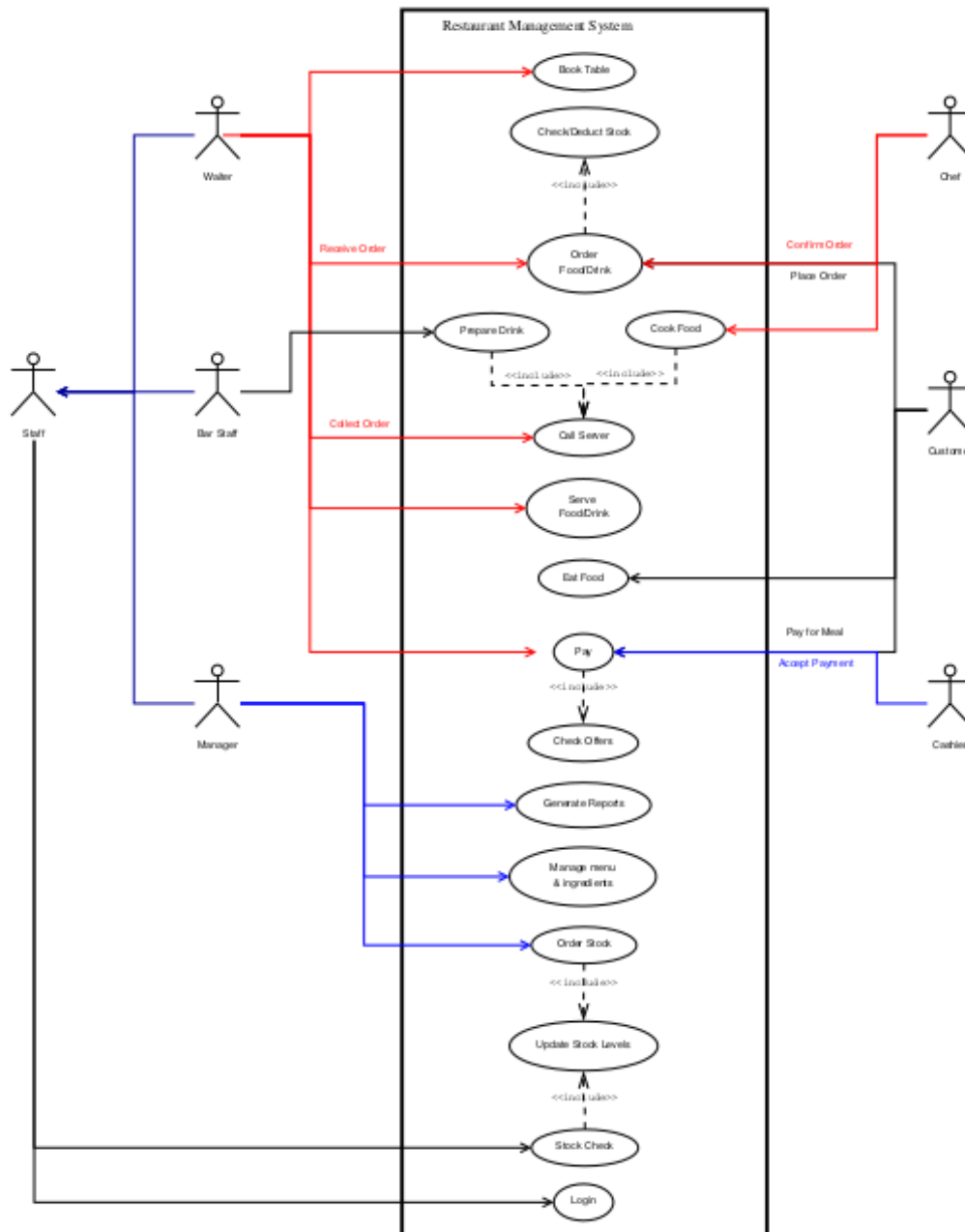


Figure 3: Use case diagram showing some of the major features within the Asian restaurant management system

The use case diagram is designed to be sequential so by following the use cases down the spine, one can follow the major steps of an order and several post features to query the data.

### 3.4 Features

An important part of requirements gathering is the production of a list of system features that categorises on priority.

Table 2: A table showing the proposed system features and allocated priorities.

<b>Feature</b>	<b>Priority</b>
Communication of data between each application	1
Meal ingredient	1
Ability to define groups of ingredients that may be used in numerous meals	2
Flexible meal grid design to fit any screen size	2
User login functionality	3
Interface for table management and selection	3

### 3.5 Measureable Goals and Requirements

The measurable goals and requirements of the system are a list of manageable requirements and goals for each application that can be prioritised and ‘ticked off’. The software requirements specification (SRS) is a complete list of requirements to be designed and developed and can take the form of functional or non-functional requirements.

### 3.6 Chapter Summary

This chapter has discussed the requirements of the system and the development methodology that will be used throughout this project. The design of the system is documented in the next chapter.



## 4 Chapter Four

### Design

#### 4.1 Chapter Overview

This chapter will focus on the design of the system using diagrams to illustrate graphically certain sections of the software system.

#### 4.2 Introduction

This project has been designed using numerous diagrammatic techniques. Recall from Section 1, that the most general modelling language to describe both the structure and behaviour of a software system is Unified Modelling language (UML). Use case diagrams have already been used in the requirements analysis as a way to graphically overview the order process within the system. Other diagrams from the UML family are used in the design stage to show the structure and behaviour of numerous sophisticated design feature.

#### 4.3 Component Diagram

A component diagram is part of UML and its main purpose is to show the structural relationship between components in the system. A component diagram is useful for this system as it shows the higher architecture, as shown in figure 4

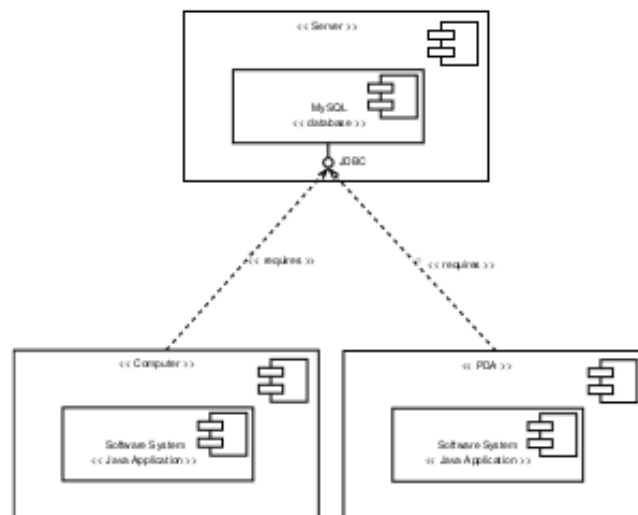


Figure 4: Component diagram showing the higher level architecture of the system.

#### 4.4 Data Storage

The restaurant management system will be built around the data storage technique therefore choosing the most appropriate persistent 1 data storage is critical to a successful

project and we can assume a flat file storage approach is inadequate. The two most popular types of persistent data storage available are relational database management system (RDBMS) and extensible markup language (XML)

#### **4.4.1 Relational Database Management System (RDBMS)**

A relational database management system (RDBMS) is a database managed system based around a relational model and are the corner stone's to many software systems including web based systems.

RDBMS are one of the most popular data storage methods out in the market and offer many advantages including:

- Fast data extraction using structured query language (SQL).
- Good management of data and security through the management system.
- Good level of data consistency.
- Advanced features including functions and triggers.
- Requirement of a data model to be developed; leading to long term cost effectiveness.

In industry, there are numerous expensive highly functional RDBMSs including Oracle and SQLServer that are very popular and offer technical support. However, there are also numerous open-source solutions with many adjudged to be as good or better and are becoming even more popular with small scale software systems.

#### **4.4.2 Extensible Markup Language (XML)**

XML is a markup language that was designed to transport and store data and is another example of a persistent data storage technique. However, it is not a predefined language thus all tags must be defined and due to its hierarchically data structure all elements must be promoted or demoted. XML could be used in two different ways in data storage; storing the XML documents within a database or having the XML documents as the fundamental unit of storage. In both cases the XML can be queried using either XPATH or XQUERY which are query languages for extracting data from XML documents.

#### **4.4.3 Storage Method Chosen**

The main difference between XML and RDBMS is that XML is hierarchical and RDBMS is relational. As restaurant data can be best represented in a hierarchical way one would believe that XML would be the best approach but it's not always that straight forward. SQL is an extremely flexible and robust querying language and for the queries required and the type of software system begin designed, it was concluded that RDBMS would be the best storage method. The next choice was to decide on the type of RDBMS to use. As discussed there are many open source RDBMSs available for us to choose from and

for the main reason of experience, MySQL was the preferred option. Figure 5 shows just how competitive the performances of different RDBMSs are.

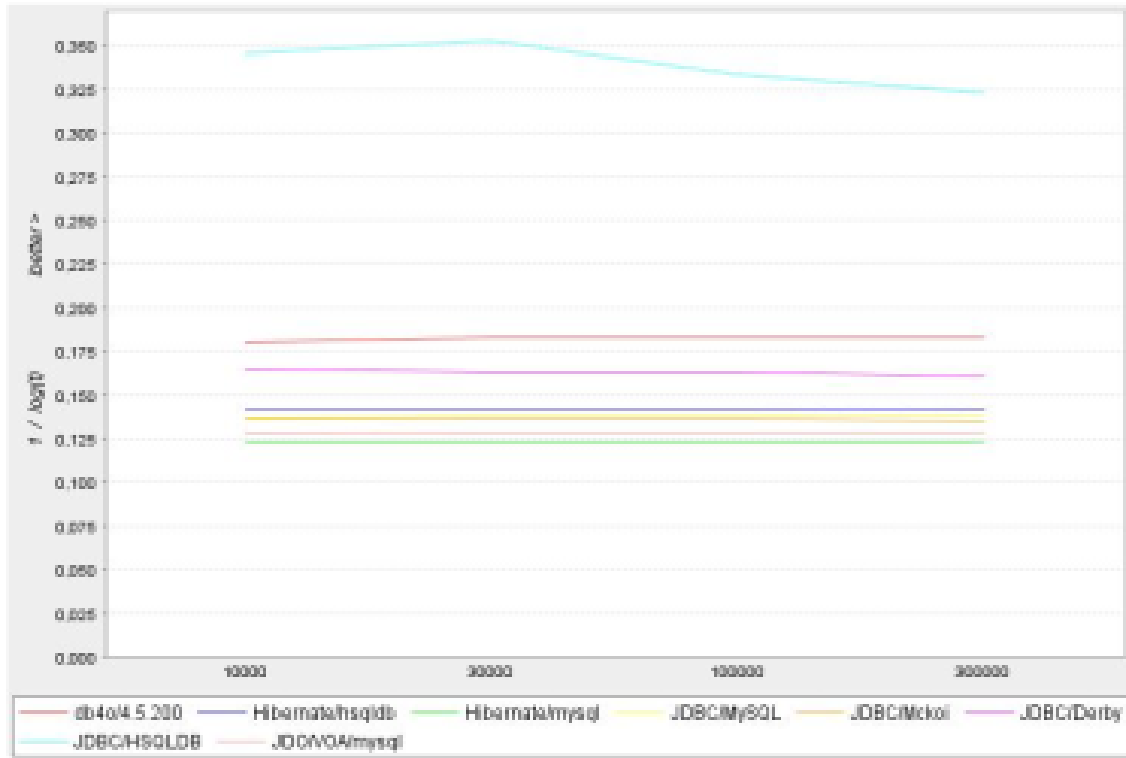


Figure 5: Database comparison diagram [2]

#### 4.4.4 Normalisation

Normalisation comes in many forms ranging from first normal form to sixth normal form. The normalisation of a database is a systematic way to free the database of undesirable characteristics where inserts, updates and deletions of data could lead to the loss of data integrity. The greater the normal form, the greater the data integrity of the database. The database in this system was designed to be in Boyce-Codd normal form which is a slightly stronger version of the third normal form. For the database to be in Boyce-Codd normal form, it had to pass for all previous normal forms as well as Boyce-Codd normal form.

A well designed database will normally abide by the first, second and third normal forms as they are the basics to a well structured relational database. According to Horsforth School [17], the first three normal forms can be defined as:

- Every attribute is atomic or single valued therefore there are no repeating fields.
- All attributes not part of the primary key must be dependent on the full key.
- There must be no transitive determinants, or each attribute that is not part of the key must be determined only by the key.

Finally for the database to be in the desired Boyce-Codd normal form, all tables must abide by the first, second and third normal forms and must not have any determinants that are not candidate keys for the table.

## **4.5 Chapter Summary**

This chapter has displayed many graphical representations of the design of the system. The implementation of the system is documented in the next chapter.

## **5 Chapter Five**

### **Implementation**

#### **5.1 Chapter Overview**

This chapter discusses the theory behind the implementation of interesting and complex features using pseudocode and diagrams for clear explanations. The implementation stage of any project can be the most enjoyable part and with a well structured plan and appropriate development methodology can be a trouble free stage.

#### **5.2 Chapter Summary**

This chapter has discussed the interesting aspects from the implementation stage. The next chapter documents the results by demonstrating the working system.

## **6 Chapter Sixth**

### **Conclusion**

#### **6.1 Chapter Overview**

This chapters draws the project report to a close and reflects on the design decisions made throughout. It also discusses possible future development ideas.

#### **6.2 Chapter Summary**

This chapter has concluded the project report and provided an insight into possible future development.

# ACKNOWLEDGEMENTS

I would like to thank my project supervisor, Prof. Liang Zhao, for providing an awful amount of guidance and input throughout the writing of this report. In addition, I'd like to thank my family for the support throughout my final year at university, and for checking over my report.

## References

- [1] C. Murphy, “Improving application quality using test-driven development (tdd),” 2009. [Online]. Available: <http://www.methodsandtools.com/archive/archive.php?id=20>
- [2] H. db, “Database comparison,” Accessed on 13 September, 2009. [Online]. Available: [http://hsqldb.org/images/imola\\_retrieve.jpg](http://hsqldb.org/images/imola_retrieve.jpg)
- [3] G. Elliot, “Iglocal business information technology: an integrated systems approach,” 2004.
- [4] A. Manifesto, “Principles behind agile manifesto,” Accessed on 20 September, 2009. [Online]. Available: <http://www.agilemanifesto.org/principles.html>
- [5] B. Dictionary, “Stakeholder,” Accessed on 2 April, 2010. [Online]. Available: <http://www.businessdictionary.com/definition/stakeholder.html>
- [6] I. Alexander, “Computing and control engineering,” April 2003.