KEMI-TORNIO UNIVERSITY OF APPLIED SCIENCES

Designing and Implementing Platform Independent Email System based on JavaMail

Dou Yong

Bachelor's thesis of the Degree Programme in Business Information Technology
Bachelor of Business Administration

TORNIO 2011

ABSTRACT

Dou, Yong. 2011. Designing and Implementing Platform Independent Email System based on JavaMail. Kemi-Tornio University of Applied Sciences. Business and Culture. Pages 47.

The objectives of the research project is to design and implement email functions by using the JavaMail Application Programming Interface. Sending a SMS message to mobiles is also discussed in this research work. Furthermore, the research work helps users understand the email system works and email functions programmed by using Java Language.

To accomplish the objective of the thesis work, the constructive method is used as the main research method. The literatures and documentation reviewed are used to gather information and knowledge about the JavaMail, which are the supplementary method to accomplish the objectives.

Unified Modeling Language is used to design the diagrams and it is defined by Meta Object Facility that is a standard for modeling. The limitations and further development are demonstrated in discussion and conclusion chapter.

The expected outcome is the functions implemented and carried out successfully by using Java language. The thesis work can be as a reference for further development.

Keywords: JavaMail API, protocols, SMS

ABBREVIATIONS

API	Application Programming Interface
IMAP	Internet Message Access Protocol
JAF	Java Activation Framework
Java SE	Java Standard Edition
Java EE	Java Enterprise Edition
MIME	Multipurpose Internet Mail Extensions
MTA	Mail Transfer Agent
MUA	Mail User Agent
NTLM	Windows NT LAN Manager
NNTP	Network News Transfer Protocol
POP	Post Office Protocol
SMTP	Simple Mail Transfer Protocol
SMS	Short Messages Service
SASL	Simple Authentication and Security Layer
TCP	Transmission Control Protocol

FIGURES

1 INTRODUCTION

Email is a method of exchanging messages between a sender and one or more recipients. With the rapid development and popularization of the Internet, network has become an important media to exchange and expand horizons. It brings convenience for people's daily lives. In a variety of network services, email plays an indispensable role with its flexible, fast and accurate characteristics in exchanging messages. To some extent, the traditional post communication has been gradually replaced by the email systems. The number of email users is growing exponentially.

According the statistics, the numbers of email accounts are more than 3 billion currently in the world. (Su 2010.) The email system has become the largest proportion of the Internet applications. Figure 1 illustrates the global number of email accounts in 2008-2013(estimated).

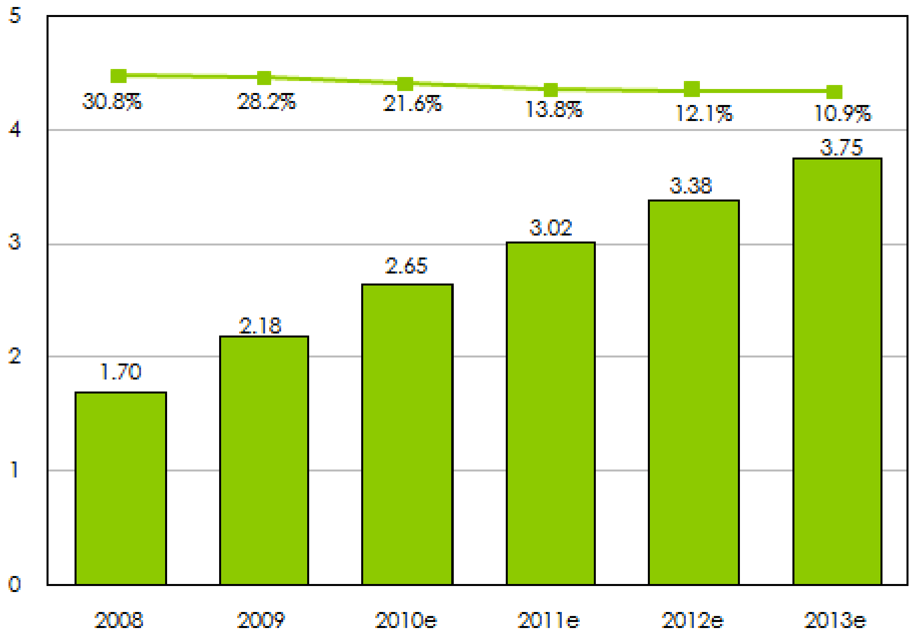


Figure 1. Email accounts in 2008 – 2013 (estimated) globally (Su 2010)

As is shown in Figure 1, the growth rate of email accounts of 2009 is 28.2% than 2008. Due to get an email account is extremely easy that promote the email widely used and popularization. An increasing number of email accounts have been registered by individuals and companies. Email can be use to send documents, pictures or videos to the recipients. Therefore, the users can share information with their friends or family by using emails. It is also a media for exchanging messages in companies.

1.1 Motivation and Purpose

Messages exchange has been an important factor for the progress of the world. People are connected with digital technology which uses to communicate with each other. Email is among the most used technologies among the modern communication technologies. Email system is easy to use for the users who are not familiar with the complexity of other modern technologies.

Today’s email systems are based on a store and transmission functions. Furthermore, most email systems depend on the Simple Mail Transfer Protocol (henceforth SMTP) to accurately transfer messages. Another necessary requirement is the user account which must be legitimate in order to send the messages through the mail server. Thus, the email server stores, receives, sends or forwards the messages.

In my thesis work, the final expected result is to implement a platform independent mail system based on the JavaMail technology. It is different from many web mail agents, such as Hotmail or Yahoo email. The email system designed is an independent application in the computer. The email users can use JavaMail to read or send mails by using their web mail accounts, if the web mail providers’ SMTP/POP servers have to be compatible with JavaMail. For the web mail agents, the browsers usually are infected by computer virus, which the operating system is broken. However, the platform independent email system has more safety than web mail agents.

Another advantage of the email system designed here is that this email system can be used to send mobile messages to cell phones. Therefore, the messages transferred between email server and mobile enhance the usable of email systems. Additionally, I am convinced that I will be benefited from my research study. Designing this email system enhances my Java programming skills during the study process. Furthermore, the email users who are interested in the principles of the JavaMail technology will get some knowledge from my thesis work.

1.2 Structure of Thesis

This study is divided into seven chapters. Chapter 2 illustrates the objectives of my thesis including research topic, research questions and expected results of the thesis. The chapter 3 focuses on research methodology and data collection. The basic information of the JavaMail technology is presented in chapter 4, which includes some related protocols and running environment. This chapter is necessary to help the users better understand the JavaMail.

In chapter 5, the requirements are depicted by using software engineering, requirements engineering and the diagrams. The requirements of the resulting system and functionality designed by using the Unified Modeling Language (henceforth UML) defined by the Meta Object Facility standard, including use case Diagram, sequence diagram, class diagram and database diagram. Chapter 6 deals with the preliminary functions are implemented by using the JavaMail. The implementation of the functions of this email system, including sending email, forwarding email, user authentication and replying email, are discussed in this chapter. Finally, chapter 7 concludes the thesis as well as gives suggestions for further development and touches upon the limitations of the research.

2 RESEARCH TOPIC, QUESTIONS AND EXPECTED RESULTS

The chapter to follow focuses on discussing research topic and questions. Further, the expected results are also included in this chapter.

“Good research is made useful by effectively communicating its purpose, methodology, results and implications. The purpose of research is to answer specific research questions and thereby enable better decision making. The results and implications summarize the major findings and conclusion as they relate to the study’s objectives.” (Hair & Money & Samouel & Page 2007, 398.)

2.1 Research Topic and Questions

The objective of my research work is to design and implement a platform independent email system based on the JavaMail technology. The email system designed includes the tools for sending emails, replying emails, receiving unread emails and forwarding emails. To achieve the core objectives of the thesis, the JavaMail knowledge needs to be acquired and the advantages of the JavaMail have to be analyzed. The whole process of the research has to be supported by the JavaMail and Java language.

Due to the fact that time is limited in the implementation phase, this research focuses on the designing system and the functions discussed above are carried out successfully. However, the user interface and other advanced functions will be designed and developed in future work.

Research questions affect the results of the research project. The research questions also help me determine the direction of the research study and better understand the core of the research work. The following research questions concerning the core of this research project are as follows:

RQ1. What does the JavaMail offer to implement a platform independent email system?

This question plays an important role in my research project. It is about the core of my research project. The whole process depends on the JavaMail Application Programming Interface (henceforth API). At beginning of the research work, I needed to learn more about the JavaMail API. The answer of this question has mainly to do with the reason why I design a platform independent email system by using the JavaMail API. If one kind of application can be run on a different operation system, it will be popularized by some potential users. I am confident that I will also benefit from the research of the compatibility between the platform independent email system and the other web mails.

RQ2. What is the architecture and functionality of the resulting system?

The answer of this question helps me achieve the objectives of this research work. The architecture and functionality of existing email systems have to be collected and analyzed. Explicit understanding the architecture and functionality built a fundamental blueprint for the designing system, and is also references for the requirements analyzed of the resulting system. The UML was used to design the architecture and functionality in my research project.

RQ3. How the solution is designed and implemented for handling SMS messages between the JavaMail and mobile phones?

The function of sending mobile messages by using the JavaMail is a feature in my research project. If the persons who the sender wants to contact are only available by their mobile phones, the sender can simply send them an email to their mobiles instead of their normal email addresses. It is convenient to exchange messages with a mobile phone via email. There are several principles about this function that the designer of this system needs to be familiar with, such as what protocols can transfer the messages from email server to a cell phone and how the Short Message Service (henceforth SMS) works.

2.2 Expected Research Results

The expected results of my research project is to implement the functions including sending emails, replying emails, forwarding emails and receiving emails by using the JavaMail API. It also can help the email users better understand how the email systems work and how to send a short message to cell phones by using email. The analysis of the JavaMail assists to use it in development work in future. Furthermore, the process of my research enhanced my study capacity, research and analysis capabilities, and programming skills. It was important to acquire the knowledge on the JavaMail API, and improve Java programming for future experiences.

2.3 Theoretical Framework

The theoretical framework is a software engineering standard that verifies the resulting output. I use the Unified Modeling Language (henceforth UML) to state the functions, process and user-system interaction in my study project. Therefore, it is necessary to choose a software engineering standard to verify the validity of the design. “The UML is the Object Management Group’s (henceforth OMG) most-used specification, and the way models application structure, behavior, and architecture. UML along with the Meta Object Facility (henceforth MOF), provide a key foundation for the OMG’s Model Driven Architecture, which unifies every step of development and integration from business modeling, through architectural and application modeling, to development, maintenance, and evolution (OMG, Inc.2011).”

The Meta Object Facility originated in the UML, the OMG was in need of metamodeling architecture to define the UML. The MOF is an international standard ISO/IEC 19502:2005 in information technology, which use to define the meta-models. (OMG, Inc.2011.) Therefore, during designing the UML diagram during my thesis object process, the MOF is used as a standard to be able to define the designing diagrams.

3 RESEARCH METHODOLOGIES

This chapter to follow discusses research method used and data collection of this research work.

“Methodology is a detailed account of the research design and the way the project is implemented. Research work and comparable development work refer to systematic activity to increase the level of knowledge to find new application. The essential criterion is whether the activity generates fundamental new knowledge (Hair et al. 2007, 398).”

3.1 Research Method

A research method defines what activity of research is, how to process, how to measure and what it takes to success. The research method used in my research work is constructive method. The constructive method was chosen because the research objective is to design and develop a new solution for a practical problem which was built based on previous example research.

Constructive method aims at producing novel solutions to practically and theoretically relevant problem. In addition, it is widely used in software engineering and computer sciences. This method also builds an artifact that solves a domain specific problem in order to create knowledge. (Ryabov 2010.) In other words, Järvinen (2001, 88) suggests that “it is typically for constructive research to build a new innovation and this process is based on existing knowledge or new technical advancement.” Furthermore, the objective of my research work is to develop a new solution for the email systems based on the JavaMail. However, most email agents are built in web browsers that have various advertisements and are easily infected by computer virus. Thus, I offered a better solution by designing a platform independent email system using the JavaMail API. The main phases of using constructive method include reviewing literature and documents about the JavaMail API, learning and analyzing principles of the JavaMail, and finding out a better solution of implementing the resulting email system for the purpose of platform independent.

3.2 Data Collection

To collect and analyze data of the JavaMail is very essential to evaluation of requirements of the designing system, and is very critical to implement functionalities. Additionally, the literature reviewed and analyzed built a theoretical foundation as well as the guideline for this research work. Data collection through related literature review and the JavaMail analyzed is the supplementary of the constructive method in this thesis work.

Data collection requires considerable knowledge and skills. The stage of the research process is very important because once the data is collected, one cannot return to an earlier step to correct decisions that led to limitation in the study. (Hair et al. 2007, 192.) At this point, the only choice is to collect the data again after correcting the problem that can be expensive and sometimes impossible.

Sekaran (2003, 104) suggests that “the type and amount of data to be collected depends on the nature of study together with research objectives.” In my research project, the data collection for the email system designed was done by analyzing software engineering, and by conducting literature review on email systems and the JavaMail technologies. The literature focused on my research objectives is formed from books, articles, journals and the Internet which helped me integrate the theoretical and practical relevance of my research study. The literature reviewed also included backgrounds of other web mail systems which are the examples of architecture well defined. “The purpose of literature review is to convey to the reader what knowledge and ideas have been established on a topic, and what their strengths and weaknesses are (Taylor 2009).”

Furthermore, to review and analyze related protocols which helps with being able to understand the knowledge of JavaMail deeply.

4 JAVAMAIL

The research project is entirely related to the JavaMail software and Java programming. Therefore, it is important to fully understand the basic knowledge and background of the research in order to be able to design a platform independent email system by using the JavaMail. This chapter discusses the characteristics of the JavaMail and in order to create a basic understanding of what the JavaMail is about.

4.1 Introduction

Email system has consistently evolved by developing innovative ideas and introducing new features which has essentially impacted on improving social technology.

Sun JavaMail was developed for the Java Development Application Programming Interface framework to provide the public a good example. The JavaMail API offers a platform independent and protocol independent model for working with protocols including IMAP, POP, SMTP, MIME, and other Internet related messaging protocols. The JavaMail API is an optional package for reading, composing, and sending electronic messages. However, the API's main purpose is not for transporting, delivering, and forwarding messages, since these are only purviews of applications, such as sending email and other Mail Transfer Agent (henceforth MTA) type program. The JavaMail API is designed to provide protocol independent access for sending and receiving messages. One key to developing highly usable and open API framework is to emphasize an abstract interface in a way that supports the existing standards and does not limit future enhancement or alternative implementation. The Java Activation Framework (henceforth JAF) also needs to be mentioned, since it is intended to unify the manner of working with the multitude of data formats, regardless of the fact that they are simple texts or extremely complex documents composed of images, audio or video. (ORACLE 2011.)

“The JavaMail API presents an excellent opportunity for developers to integrate advanced mail capabilities into the applications with all benefits that the JavaMail can provide, such as ease of development, multiple protocols support, and flexible integration of future advances (Nemil & Hemrajani 1999).” That is the reason the JavaMail is chosen to design and implement the email system here, despite there are several Mail User Agents existed. The daily email program is designed to be read and written with MTAs. However MUAs rely on MTAs processing to delivery the mail.

Figure 2 depicts an email with recipient’s address yong@hotmail.com is sent to DNS server via SMTP server. The DNS server seeks out the domain name of recipient is the hotmail.com, and sends this email to hotmail MTA server. The MTA server finds out the yong is the user account, and deliveries the email to the recipient.

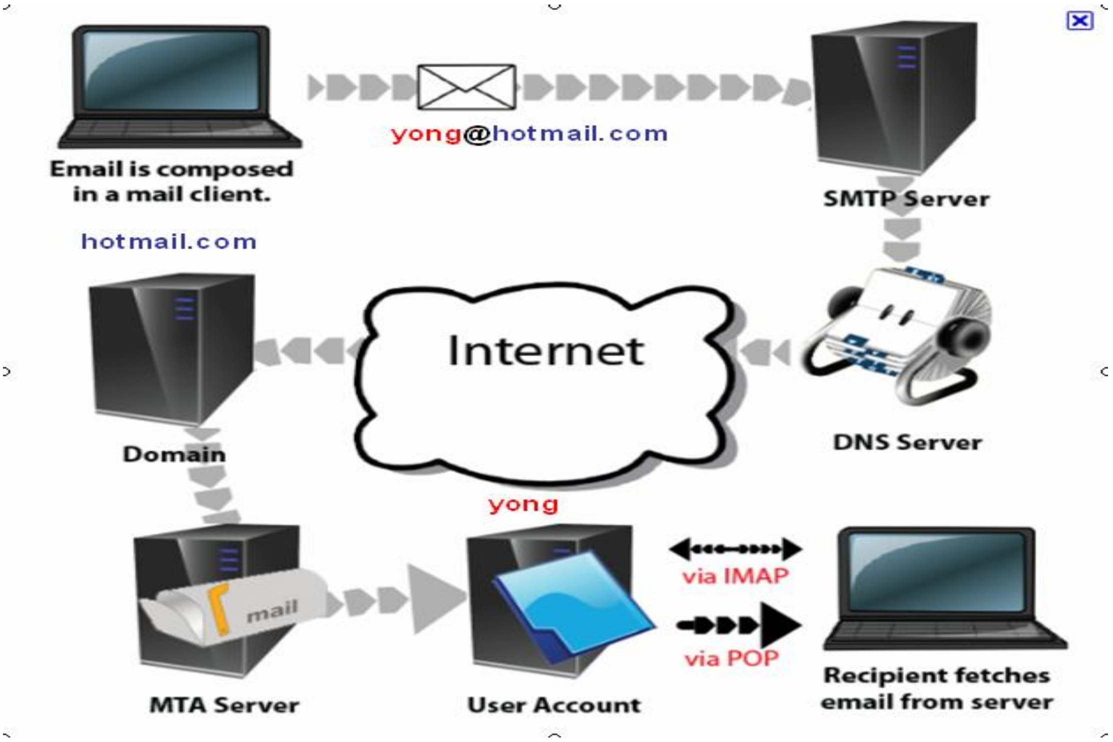


Figure 2. Email system works (Trivedi 2011)

4.2 Review of Related Protocols

As was discussed above there are four basic protocols that help better understand the JavaMail. Further, there are other protocols supported by the JavaMail such as Network News Transfer Protocol (henceforth NNTP). The following information will introduce the main protocols that are widely used in email systems.

4.2.1 Simple Mail Transfer Protocol (SMTP)

The Simple Mail transfer Protocol defines the mechanism for delivering email. The SMTP is a protocol for sending email messages between email servers over Internet and network. It also sends and authenticates a username and password to an email server. If users' information is not correct, the users are not allowed to go beyond email server. (Snoke 1999.)

4.2.2 Post Office Protocol (POP)

POP stands for Post Office Protocol. Currently POP3 is popular in mail system. The POP3 provides a mechanism that people on the Internet use to get their mails. It defines support for a single mailbox for each user. Most people are familiar with using POP3, however there are not all features are supported, such as the ability to see how many new mail messages they have. (jGuru 2001.)

4.2.3 Internet Message Access Protocol (IMAP)

IMAP stands for Internet Message Access Protocol. It is a more advanced protocol for receiving messages than POP. Cyberindian (2008) suggests that "it was designed to overcome some problems with POP behavior and provides more features for delivery and management of email." The IMAP service provides folders for the user to store emails and attachments on the server. Assuming one's mail server supports the IMAP, the JavaMail based program can take advantage of users having multiple folders on the server and these folders can be shared by multiple users. Compare with POP, the advanced capability of the IMAP is obvious. Due to the more advanced capabilities, the IMAP places a much heavier burden on the mail server, requiring the server to receive the new messages, deliver them to users when requested, and maintain them in multiple folders for each user. (Zukowski 2000.)

4.2.4 Multipurpose Internet Mail Extensions (MIME)

Multipurpose Internet Mail Extension (henceforth MIME) is not a mail transfer protocol. The MIME defines the content of what is transferred, the format of the messages, attachments and so on. Multipurpose Internet Mail Extension supports the following formats of email:

- Text in character sets other than ASCII
- Non-text attachment
- Message bodies with multiple parts
- Header information in non-ASCII character sets. (Freed & Borenstein 1993.)

In the JavaMail API, the users usually do not need to concern about these formats.

4.3 Environment Supported

“The JavaMail API is available as an optional package for using with Java Standard Edition (henceforth SE) platform and is also included in Java Enterprise Edition (henceforth EE) platform.” To implement the results of my research project, Java programming language and Java platform are fundamental technologies. The Java programming language is a high level object-oriented language. The Java platform is a specific environment in which Java programming language application runs. All Java platforms consists a Java Virtual Machine (henceforth VM) and an application programming interface (henceforth API). The Java VM is a program for a specific hardware and software platform that run Java applications. An API is a collection of software components that one can use to create other software components. (ORACLE 2011.)

4.3.1 Java Standard Edition (Java SE)

Java SE is the most popular and widely used. It provides the core functionality of the Java programming language. It also defines everything from the basic type and objects. Java SE lets users develop a Java application on desktops and servers, as well as demanding embedded and real-time environments. (ORACLE 2011.)

4.3.2 Java Enterprise Edition (Java EE)

Java EE is built on top of the Java SE platform. "The Java EE platform provides an API and runtime environment for developing and running large-scale, multi-tiered, reliable and secure network application." For the enterprise application, Java EE is designed to the complexity of enterprise application by supplying a development model and API that allows developers to focus on functionality. For the tiered application, Java EE development concentrates on the middle tier to make enterprise application management easier, and more secure. (ORACLE 2011.)

5 REQUIREMENTS ANALYSIS AND FUNCTIONALITY DESIGN

This chapter concentrates on formulating the system designing requirements analysis, and software requirements analysis. I will use the UML diagram to discuss the architecture and functionality of the system.

5.1 System Requirements Analysis

“Requirements are a predefined set of system functions, attributes and development stages as a specification of what should be accomplished during the development process.” These requirements describe the system behavior and implementation phase of a system. (Sommerville & Sawer 1997, 4.) Requirements also illustrate what kind of features and functions are needed for a specific research. Therefore, it is necessary to define and analyze the system requirements at the beginning of the designing system.

In system engineering, a requirement can be a description of what a system must do, referred to as a Functional Requirement. It captures the intended behavior of the system. This behavior may be expressed as services, tasks or functions the system is required to perform. (Bredemeyer & Malan 2001.) In my study project, the functional requirements rely on user requirements. Therefore it is important that design a system in order to be able to achieve what the users actually want it to be. Clear understanding the user requirements is a part of the resulting system designed and is critical to the success of an interactive system. A successful system begins with an understanding of the needs of the users. In other words, how the users’ requirements can be provided by the system is considered to be as a standard on how successfully the system is performed. The method for analyzing user requirements includes the following 4 elements as is illustrated in Figure 3, including information gathering, user needs identification, envisioning and evaluation of the user needs, and final requirements specification decision.

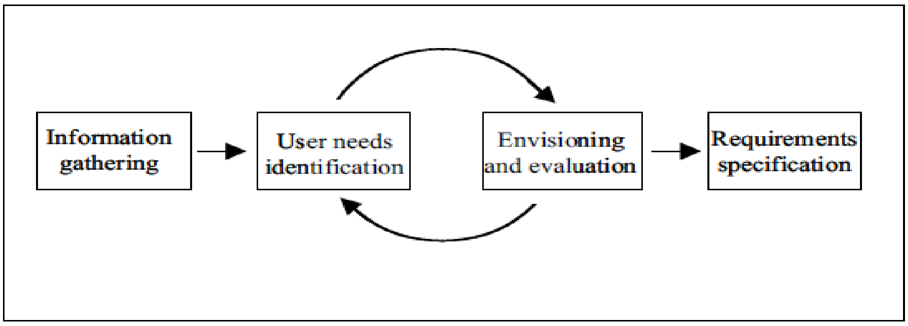

Information gathering
User needs identification
Envisioning and evaluation
Requirements specification

Figure 3. Analyzing user requirement process (Maguire & Bevan 2002.)

Therefore, it is necessary to design the user requirements for the purposes that determine the success of the resulting system. User requirements are essential to illustrate the usability in order to be able to design functional requirements for the designing system.

Another type of requirements refers to some features about the system itself, and how well it performs its function. These requirements are often called non-functional requirements. Non-functional requirements define the overall qualities or attributes of the resulting system. They also place restrictions on the system being developed, the development process, and specify the external constrains that the system must meet. Non-functional requirements include safety, security, usability, reliability and performance requirements. (Kotonya & Sommerville 1998, 11.)

Usability and security are most important in my study project. The objective of my research is to design a platform independent email system. The usability of my system is aimed to be user-friendlier than other web mail systems. There are three primary elements of usability that need to be taken into account when designing system interfaces and user-system interaction as listed below:

- Efficiency: take less time to accomplish a specific task.
- Learnablity: operation can be easier to learn by simple method.
- Satisfaction: more satisfied by most users. (Nielsen 2003.)

Therefore, how useful the system can be performed represents how readily it is accepted by users.

The email systems are established as a prime method of communication for companies' or individuals' use that is quicker and cheaper than other traditional methods. The security of my resulting system is essential as well as usability. To prevent unauthorized users from by passing the registration interface to enter into the application system directly, an authentication function was designed for the resulting system here.

5.2 Architecture and Functionality of System

The architecture and functionality of the system representation was carried out by using the UML. The UML is a visual language that can be used in developing software system. It is also a formal specification language. Rational Software Corporation and the Object Management Group (OMG) have brought together elements of three significant object-oriented diagramming notations and aspects of many other notations to produce a standard modeling language (Bennett & Skelton & Lunn 2001, 6).

There are 5 kinds of diagrams to depict the functionality of the resulting system in following paragraphs.

5.2.1Use Case Diagram

Use Case Diagrams provide a good way of getting an overall picture of what is happening in the existing system or is planned to happen in the new system. Bennett et al. (2001, 25) expound “the use case is an effective means of communicating with users and other stakeholders about the system and what it intended to do.”

The following questions should be answered before using use case diagram:

- Who are the people who will use this system to enter information?
- Who are the people who will use this system as recipients of information?
- What functions included in the system?
- What are the other systems that this system will interact with? (Bennett et al. 2001, 36.)

On the basis of the answers to the questions above, the use case diagram of my research object is demonstrated by Figure 4.

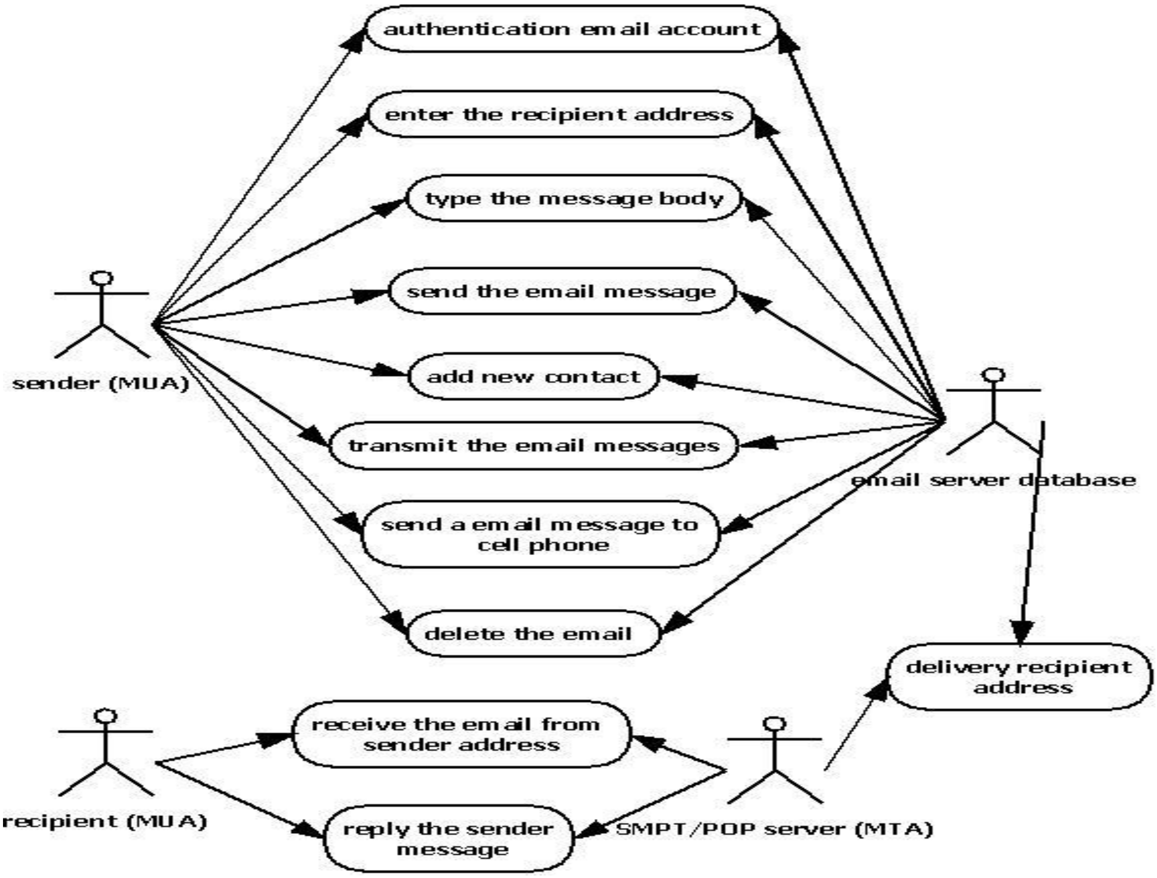

authentication email account
enter the recipient address
type the message body
send the email message
add new contact
transmit the email messages
send a email message to cell phone
delete the email
sender (MUA)
email server database
delivery recipient address
receive the email from sender address
reply the sender message
recipient (MUA)
SMPT/POP server (MTA)

Figure 4. Use case diagram of the resulting system

The Figure 4 depicts the functionality of the email system among the sender, recipient, and email server. The functionality includes authenticate email account, send email, add new contact and delete email, which are commands from user-interface to be delivered to the email server. These commands are carried out by email server, i.e. sending messages are sent to the recipients' address by smtp server.

5.2.2 Activity Diagram

The main reason to use activity diagram is to model the workflows behind the system being designed. Activity diagram is also used to analyze a use case by describing what actions need to be taken place and when they should occur, depict a complicated sequential algorithm, and model applications with parallel processes. (Braun & Sivils & Shapiro & Versteegh 2001.)

Bennett et al. (2001, 225) describe the method to be produced activity diagram for system use case modeling:

- Identifying key scenarios of system use case, using primary and alternative paths.
- Combining the scenarios to produce comprehensive workflows described using activity diagram.
- Where significant object behavior is triggered by a workflow, adding object flows to the diagram.
- Where workflows cross technology boundaries, using swim lanes to map the activities.

According these steps above, the activity diagram of my study object can be drawn as following Figure 5:

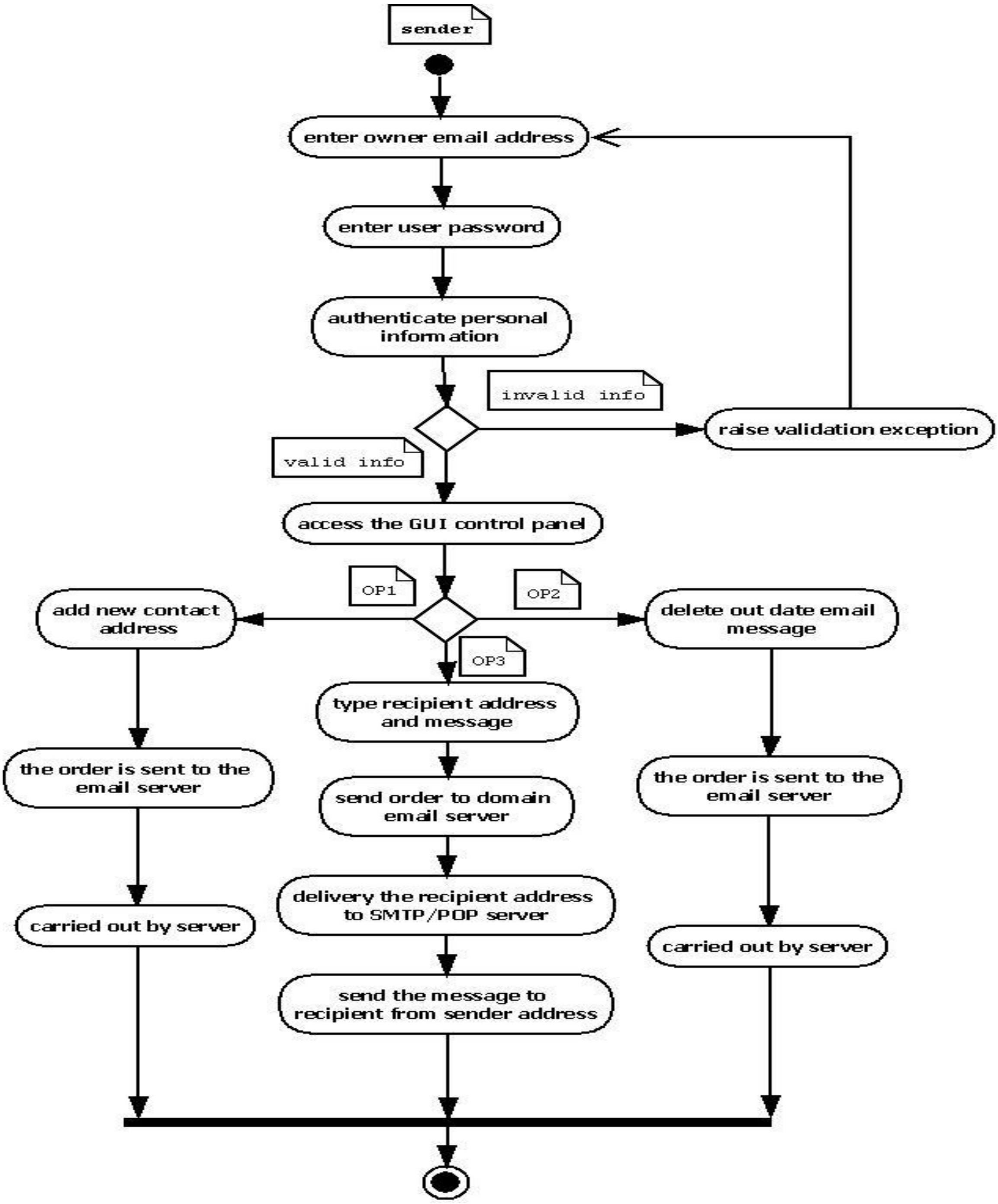

sender
enter owner email address
enter user password
authenticate personal information
invalid info
raise validation exception
valid info
access the GUI control panel
OP1
OP2
OP3
add new contact address
delete out date email message
type recipient address and message
the order is sent to the email server
send order to domain email server
the order is sent to the email server
delivery the recipient address to SMTP/POP server
carried out by server
carried out by server
send the message to recipient from sender address

Figure 5. Activity diagram of the resulting system

The Figure 5 is an activity diagram to illustrate the resulting system. The sender's address and password are authenticated by email server. If they are correct, the sender is allowed to access into the email system, otherwise the sender has to be re-authenticated. There are three option activities selected to explain the functionality from start to end, including typing recipients' address and messages, delete out date email system, and add new contact addresses.

5.2.3 Sequence Diagram

Sequence diagrams are the most popular UML artifact for dynamic modeling, which focuses on identifying the behavior within the system. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the system and the sequence of messages exchanged between the objects are needed to carry out the functionality of the system. (Bennett et al. 2001, 179-206.)

The steps in drawing sequence diagram:

- Decide on the context of the interaction: system, subsystem, use-case or operation.
- Identify the structural elements necessary to carry out the functionality of the use case or operation.
- Consider the related situation that may be required. (Bennett et al. 2001, 192.)

Sequence diagram of the resulting system is stated by Figure 6:

There are 5 objects defined in the sequence diagram. The objects include sender, GUI, email server database, SMTP/POP server and recipient. The sequence diagram describes the actions with the time runs from left to right. The sender’s account and password are also needed to authenticate by email server database. If the account is valid, the actions include enter recipient address, type message body, delete out date email and add new contact are carried out by email server database. With the time running, the message is sent to the recipient address through SMTP/POP server.

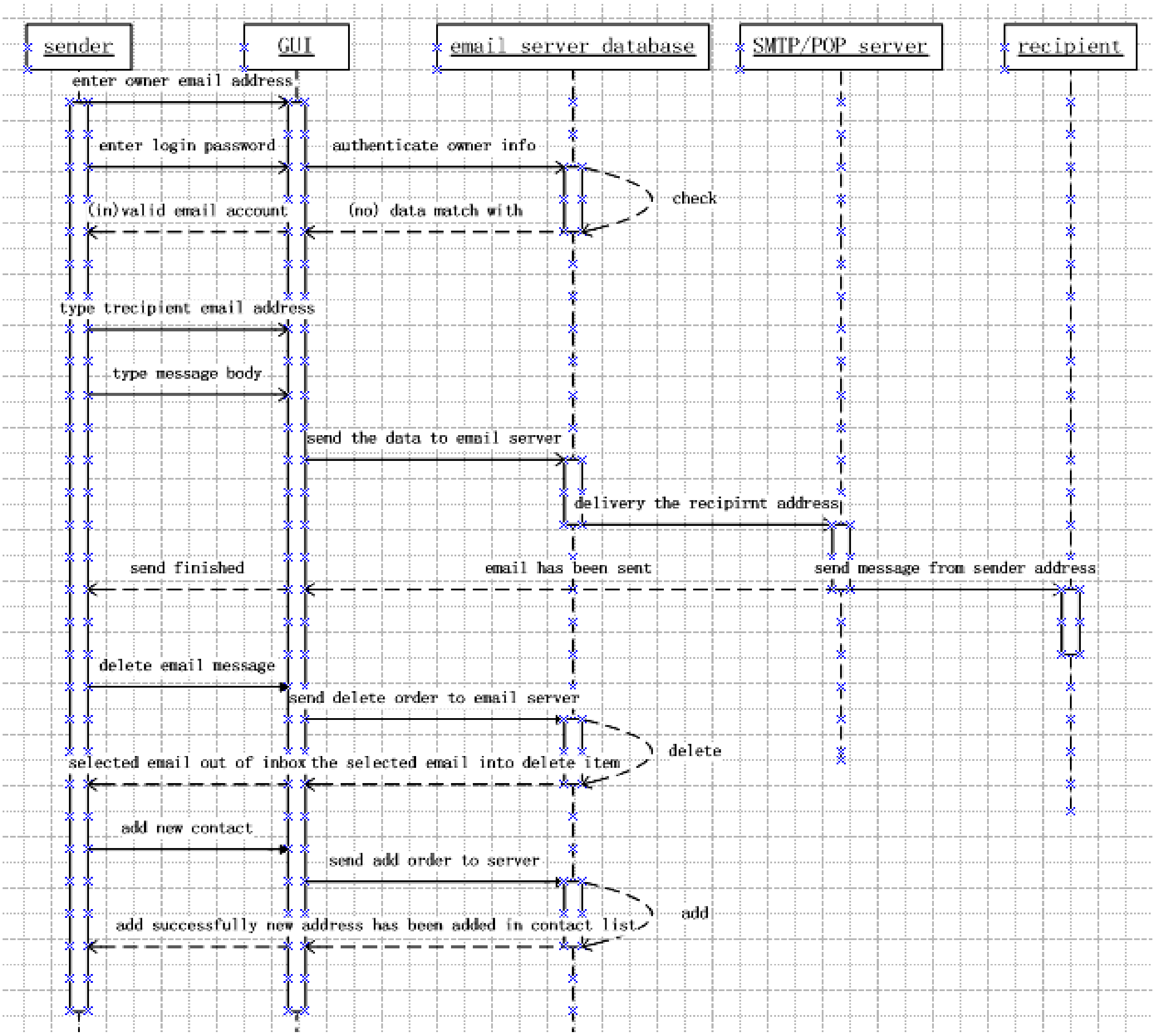

sender
GUI
email server database
SMTP/POP server
recipient
enter owner email address
enter login password
authenticate owner info
check
(in)valid email account
(no) data match with
type trecipient email address
type message body
send the data to email server
delivery the recipirnt address
send finished
email has been sent
send message from sender address
delete email message
send delete order to email server
delete
selected email out of inbox
the selected email into delete item
add new contact
send add order to server
add
add successfully
new address has been added in contact list

Figure 6. Sequence diagram of the resulting system

5.2.4 Database Diagram

The database of the resulting system was divided into 4 parts, including user-account, send-server, receive-server and email-content. Designing the send-server and receive-server respectively is better understood by the reader.

The database diagram of system is following Figure 7.

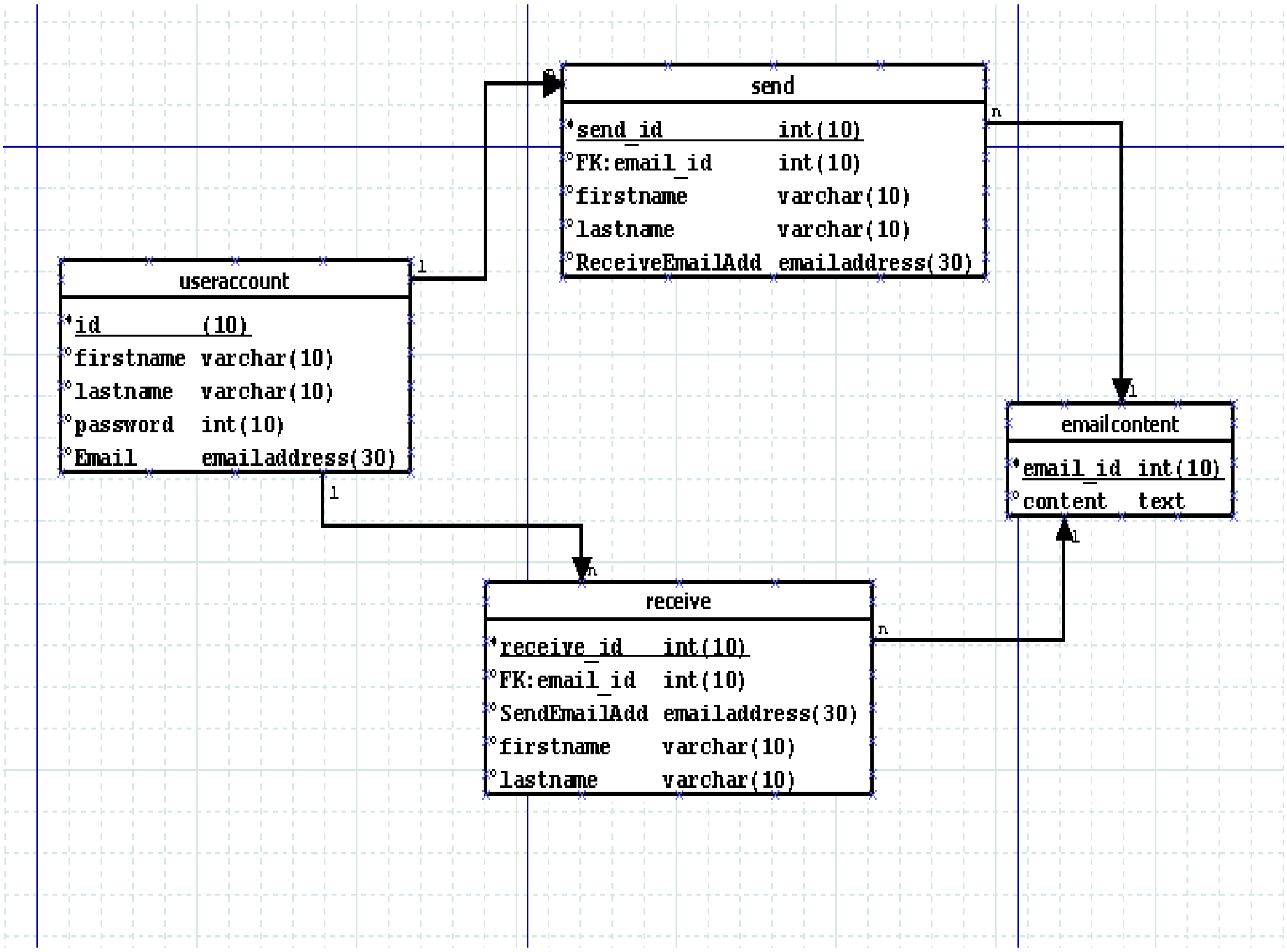


Figure 7. Database diagram of the resulting system

The user account database includes firstname, lastname, password and email addresses, which is used to authenticate the host and store recipient's data. The send and receive database store the senders/recipients' data respectively. The content of email is stored separately by email content database.

5.2.5 Class Diagram

A class diagram shows a static view of the classes in a model. The attributes and operations of classes are shown along with various different relationships that bind the classes together. The class diagram forms a necessary and fundamental part of any object-oriented model produced using the UML. It shows the behavioral and data management connection of each class and how these connections are assigned across the classes. Conversely, they neither show the functional requirements of a system from the expectation of the end-users and nor show how different elements of a class model interact with each other. (Bennett et al. 2001, 47.)

The main purpose of using class diagram:

- Describe the associations, generalization and aggregation relationships among classes.
- Show the features of classes, principally the attributes and operation of each other.
- Used throughout the development lifecycle.
- Show individual object instances within the class structure. (Bennett et al. 2001, 47-75.)

The class diagram of resulting system is Figure 8:

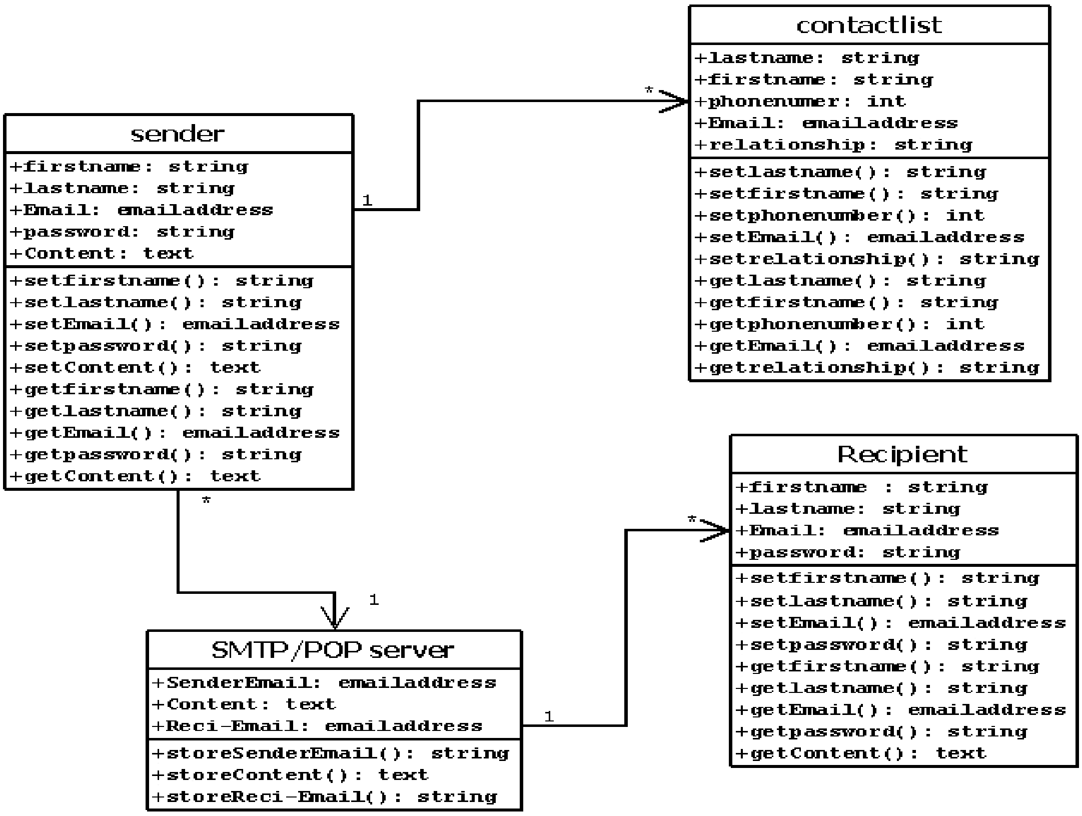


Figure 8. Class diagram of the resulting system

6 IMPLEMENTATION BASED JAVAMAIL

This chapter focuses on the implementation of the resulting system. It includes JavaMail installation, environment for implementation and functionality implementation. The answer of sending a message to cell phone will be found from this chapter.

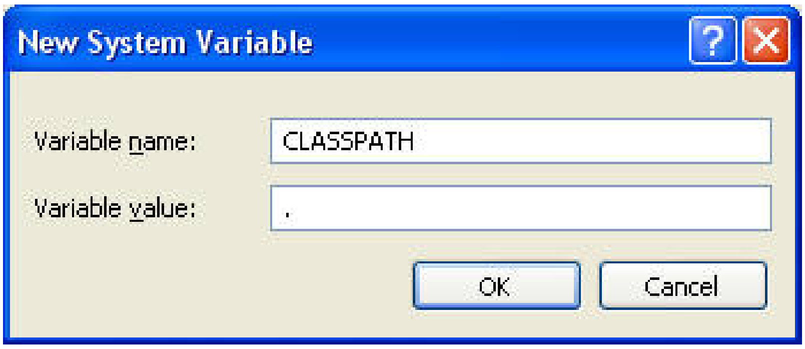
6.1 JavaMail Installation

The latest version of JavaMail is 1.4.4 and it was released on January 21, 2011. It includes the following new features:

- Ability to cache POP3 messages on disk.
- In memory POP3 messages cache now use soft references.
- NTLM authentication support is now integrated, no longer need jcifs.jar.
- SASL authentication support for SMTP.
- New demo classes showing how to handle old non-MIME Outlook messages. (ORACLE 2011.)

To use the JavaMail 1.4.4 API, the JavaMail implementation file is necessary, unzipped the javamail 1-4-4.zip file, and add mail.jar file into the CLASSPATH. Except the JavaMail file, the JavaBeans Activation Framework (henceforth JAF) is also required. The jaf1-1-1.zip file also needed to be unzipped, and add the activation.jar file to the CLASSPATH. The CLASSPATH has to be set up as a new system variable in Environment Variables of MyComputer properties.

Open the properties of MyComputer, there are User Variables and System Variables in the Environment Variables of advanced option. Set a new system variable as is shown in Figure 9 and the name is CLASSPATH. The value needs to be filled dot at the beginning.



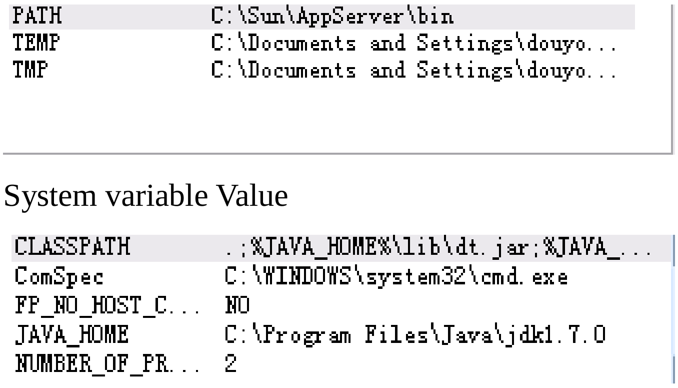
Variable name: CLASSPATH

Variable value: .

Figure 9. Set up a new variable in Environment Variable

The value of the CLASSPATH is displayed in Figure 10. .;%JAVA_HOME%\lib\dt.jar;%JAVA_HOME%\lib\tools.jar;mail.jar;activation.jar was added in CLASSPATH value.

User variables: Value:



PATH	C:\Sun\AppServer\bin
TEMP	C:\Documents and Settings\douyo...
TMP	C:\Documents and Settings\douyo...

System variable Value

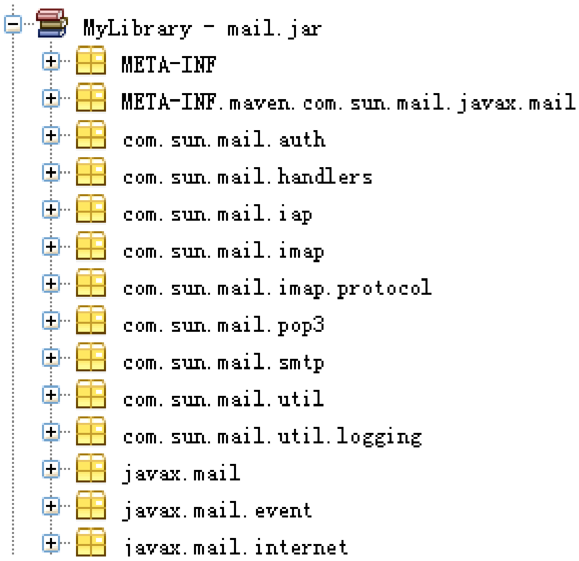
CLASSPATH	.;%JAVA_HOME%\lib\dt.jar;%JAVA_...
ComSpec	C:\WINDOWS\system32\cmd.exe
FP_NO_HOST_C...	NO
JAVA_HOME	C:\Program Files\Java\jdk1.7.0
NUMBER_OF_PR...	2

Figure 10. Add mail.jar file and activation.jar file to the CLASSPATH

6.2 Environment for Implementation

The NetBean IDE is as a Java platform to develop Java applications. It is an award-winning integrated development environment available for various operating systems, such as Windows, Mac and Linux. The NetBean project consists of an open source IDE and an application platform that enable developers create web, desktop and mobile applications using the Java platform. (NetBean 2011.)

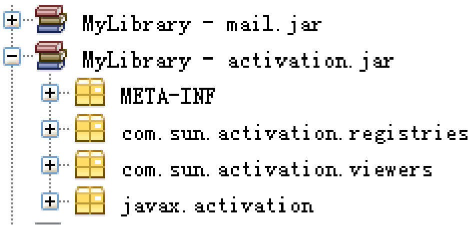
Before I start to compile the functions using Java language, the JavaMail and JAF files have to be added into the library of a specified project. To create a new library named MyLibrary, and import the mail.jar and activation.jar files into MyLibrary. The following pictures display some resources included in mail.jar and activation.jar.



```
MyLibrary - mail.jar
    META-INF
    META-INF.maven.com.sun.mail.javax.mail
    com.sun.mail.auth
    com.sun.mail.handlers
    com.sun.mail.iap
    com.sun.mail.imap
    com.sun.mail.imap.protocol
    com.sun.mail.pop3
    com.sun.mail.smtp
    com.sun.mail.util
    com.sun.mail.util.logging
    javax.mail
    javax.mail.event
    javax.mail.internet
```

Figure 11. Mail. jar resources

There are some mail protocols supported in the JavaMail package, such as imap, pop3 and smtp. The mail account authenticate is also included in the package, which used to validate the email users' account.



```
MyLibrary - mail.jar
MyLibrary - activation.jar
    META-INF
    com.sun.activation.registries
    com.sun.activation.viewers
    javax.activation
```

Figure 12. Activation. jar resources

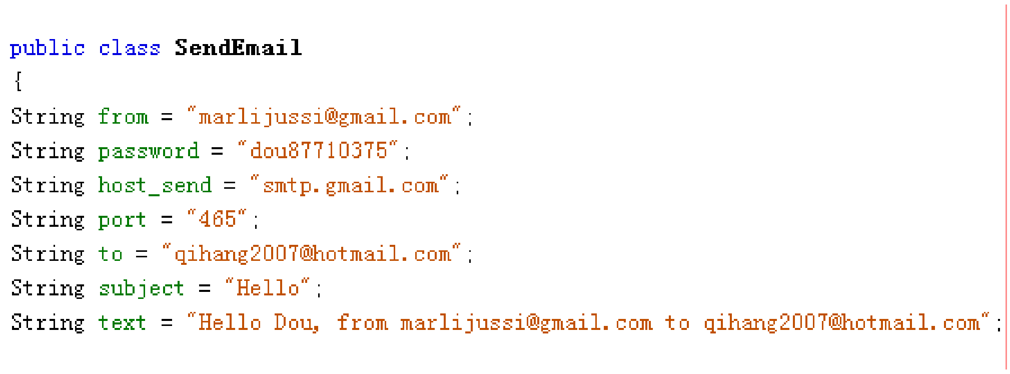
Figure 12 shows a standard extension, which unify the manner of working with the multitude of data formats.

6.3 Functionality Implementation

This module describes functionality implemented by using the JavaMail package and Java language. The functions include sending an email, receiving an unread email, replying an email and forwarding an email. Due to time limitation, the user interface is not designed and implemented in my thesis work. However, the resulting system will be improved in future study.

6.3.1 Sending an email

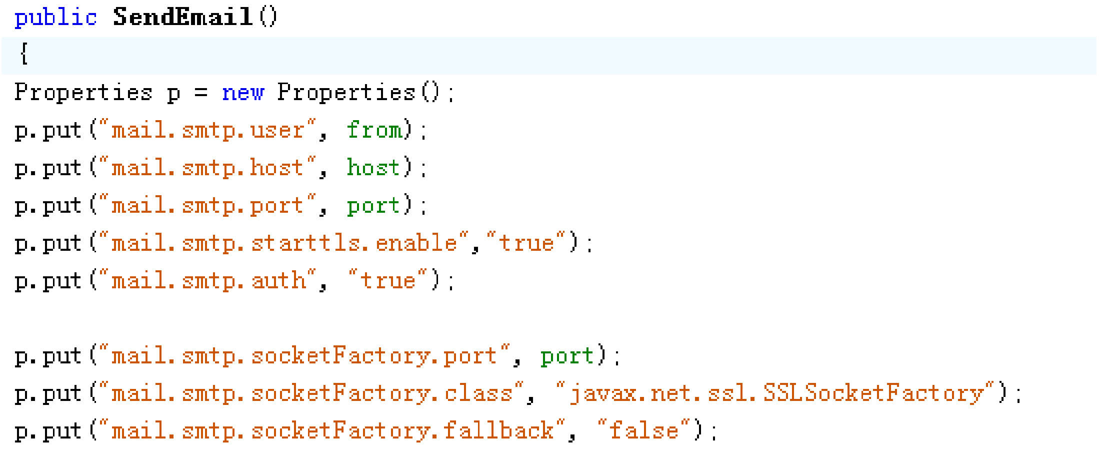
Sending an email message involves getting a session, creating and filling message body, and sending it successful. The authenticator is provided by the JavaMail API. The following picture defines some variables of sending programming.



```
public class SendEmail
{
String from = “marlijussi@gmail.com”;
String password = “dou87710375”;
String host_send = “smtp.gmail.com”;
String port = “465”;
String to = “qihang2007@hotmail.com”;
String subject = “Hello”;
String text = “Hello Dou, from marlijussi@gmail.com to qihang2007@hotmail.com”;
```

Figure 13. Define the variables of sending programming

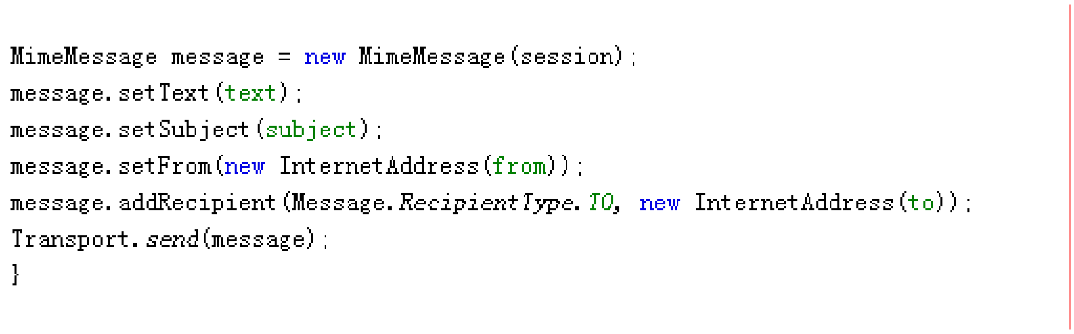
There are some variables need to be defined, including host email account, host password, host send_server used, and the port of smtp server are displayed in Figure 13. The host email and password need to be authenticated to be able to access the host email system. My Gmail account is as an example. Furthermore, the port is necessary to know for accessing the Gmail smtp server. An email is sent from my gmail account to my hotmail account.



```
public SendEmail()
{
Properties p = new Properties();
p.put("mail.smtp.user", from);
p.put("mail.smtp.host", host);
p.put("mail.smtp.port", port);
p.put("mail.smtp.starttls.enable","true");
p.put("mail.smtp.auth", "true");

p.put("mail.smtp.socketFactory.port", port);
p.put("mail.smtp.socketFactory.class", "javax.net.ssl.SSLSocketFactory");
p.put("mail.smtp.socketFactory.fallback", "false");
```

Figure 14. Setup the smtp mail server

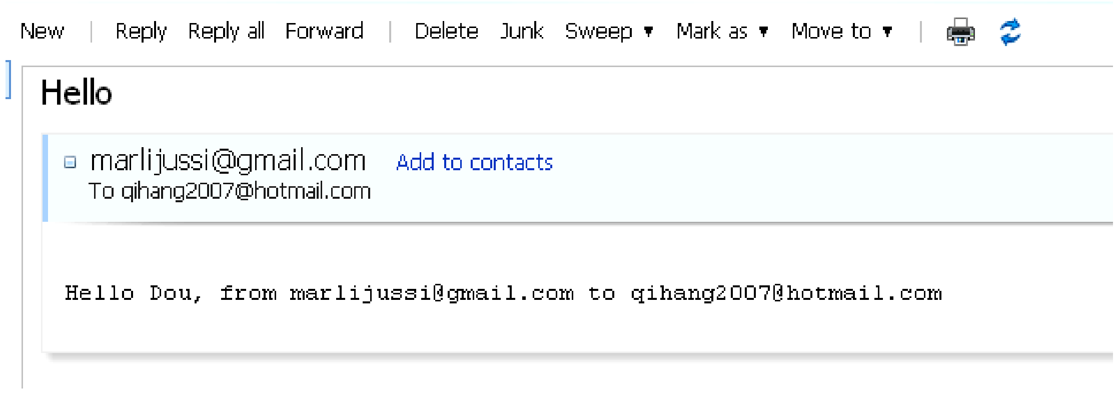
Figure 14 illustrates the programming gets system properties, uses host email account and port to access the smtp server. Email server often uses Secure Sockets Layer (henceforth SSL) to validate the email users' information. SSL is able to encrypt the important information during online exchanging.



```
MimeMessage message = new MimeMessage(session);
message.setText(text);
message.setSubject(subject);
message.setFrom(new InternetAddress(from));
message.addRecipient(Message.RecipientType.TO, new InternetAddress(to));
Transport.send(message);
}
```

Figure 15. Define subject and message body

Figure 15 defines subject and message body of an email. The sending function is implemented using "Transport. send" from host email address to recipient's address.



Hello

marlijussi@gmail.com

To qihang2007@hotmail.com

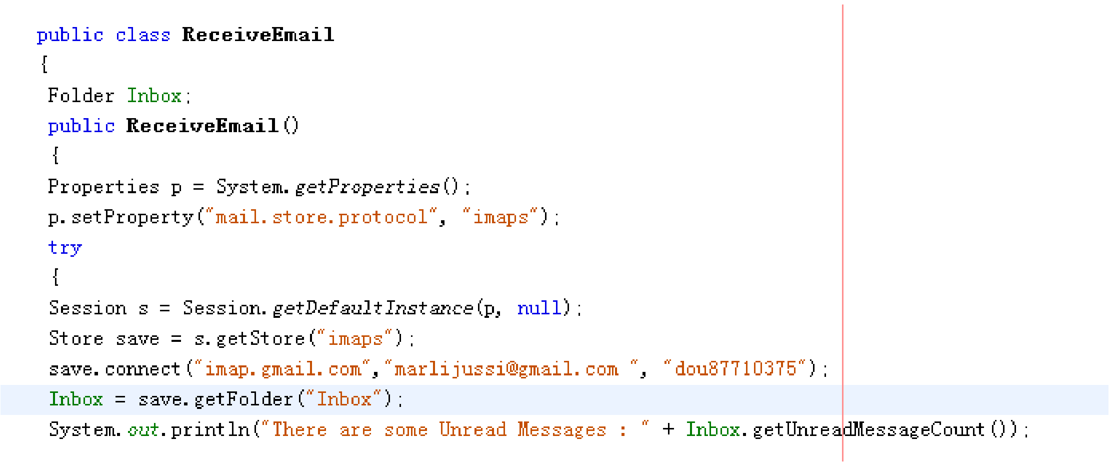
Hello Dou, from marlijussi@gmail.com to qihang2007@hotmail.com

Figure 16. The result of sending function

The Figure 16 is the result of sending message from marlijussi@gmail.com to qihang2007@hotmail.com.

6.3.2 Receiving an unread email

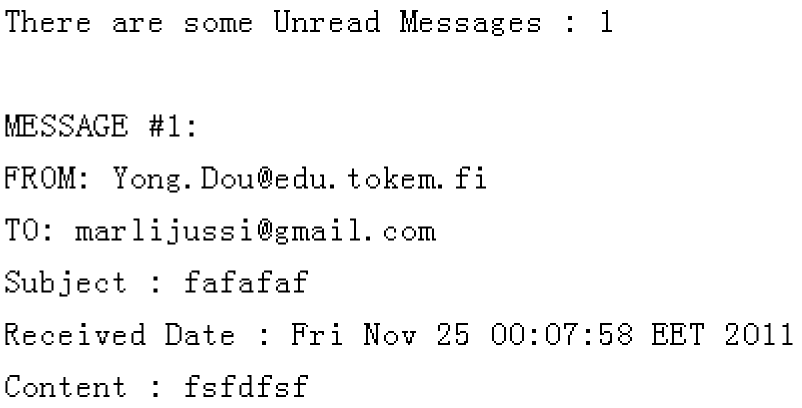
The library imported of receiving email is different with sending email. To open unread email message from host account need to use imap protocol provided by Gmail imap server. The host account also has to be authenticated.



```
public class ReceiveEmail
{
 Folder Inbox;
 public ReceiveEmail()
 {
 Properties p = System.getProperties();
 p.setProperty(“mail.store.protocol”, “imaps”);
 try
 {
 Session s = Session.getDefaultInstance(p, null);
 Store save = s.getStore(“imaps”);
 save.connect(“imap.gmail.com”,“marlijussi@gmail.com ”, “dou87710375”);
 Inbox = save.getFolder(“Inbox”);
 System.out.println(“There are some Unread Messages : ” + Inbox.getUnreadMessageCount());
```

Figure 17. Setup the imap mail server

Figure 17 demonstrates the imap mail server is setup and accessed by using the host account. The imap protocol is used in my thesis project. However the pop3 is also able to get mails from the host account. Receiving unread mail need to get inbox from mail server, hence the unread message count also has to be get from server.



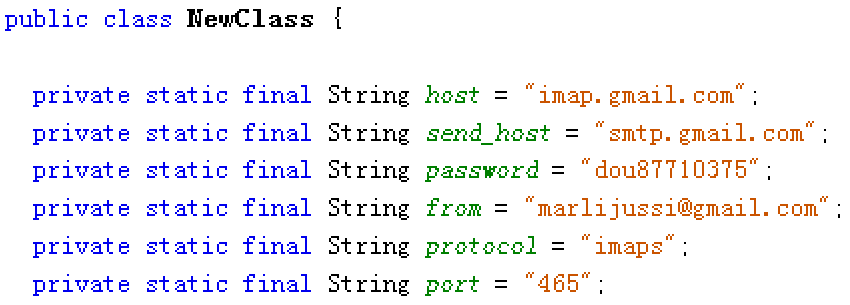
```
There are some Unread Messages : 1

MESSAGE #1:
FROM: Yong.Dou@edu.tokem.fi
TO: marlijussi@gmail.com
Subject : fafafaf
Received Date : Fri Nov 25 00:07:58 EET 2011
Content : fsfdfsf
```

Figure 18. The result of receiving email

Figure 18 shows an unread message from my Gmail account server. The sender's address, recipient's address, subject, receive date and content were needed to display in the result.

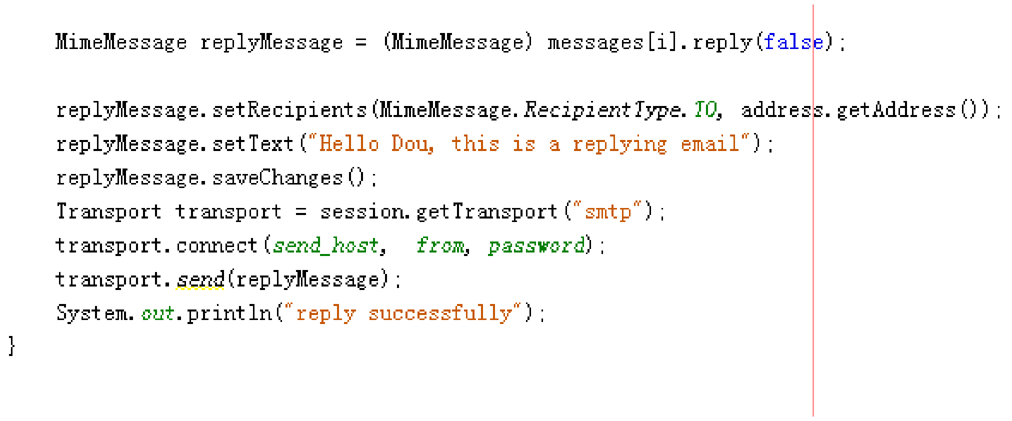
6.3.3 Replying an email

To reply an email has to need both imap and smtp protocols. The imap is used to access into the host email account and smtp is used to send a replying email to the sender. The following Figure 19 displays both imap and smtp servers have to be defined at the beginning of programming.



```
public class NewClass {

  private static final String host = "imap.gmail.com";
  private static final String send_host = "smtp.gmail.com";
  private static final String password = "dou87710375";
  private static final String from = "marlijussi@gmail.com";
  private static final String protocol = "imaps";
  private static final String port = "465";
```

Figure 19. Define the imap & smtp server



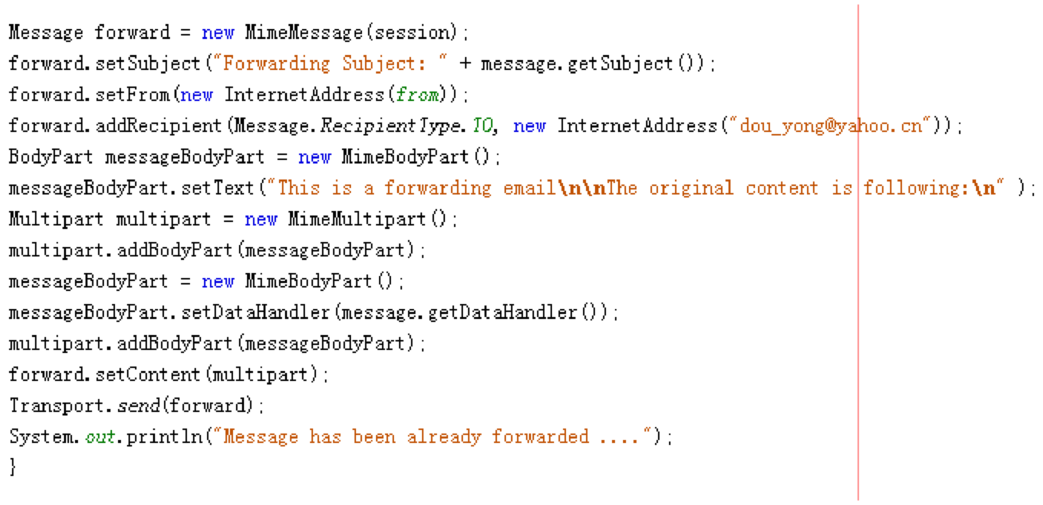
```
        MimeMessage replyMessage = (MimeMessage) messages[i].reply(false);

        replyMessage.setRecipients(MimeMessage.RecipientType.TO, address.getAddress());
        replyMessage.setText("Hello Dou, this is a replying email");
        replyMessage.saveChanges();
        Transport transport = session.getTransport("smtp");
        transport.connect(send_host,  from, password);
        transport.send(replyMessage);
        System.out.println("reply successfully");
    }
```

Figure 20. Function of replying an email

Figure 20 shows the replying function is implemented by using the “replyMessage” of the JavaMail API provided. The variables including send_host, from and password are used to connect the smtp server. If the programming of the replying function is carried out successfully, the system prints out “reply successfully”.

6.3.4 Forwarding an email

Both the smtp and imap servers are necessary for forwarding an email. To forward an email message, one part has to be created for the text of the host message and a second part for the original message. Moreover, the first part and second part need to be combined into a multipart which can be sent to a specific recipient’s address.



```
Message forward = new MimeMessage(session);
forward.setSubject("Forwarding Subject: " + message.getSubject());
forward.setFrom(new InternetAddress(from));
forward.addRecipient(Message.RecipientType.TO, new InternetAddress("dou_yong@yahoo.cn"));
BodyPart messageBodyPart = new MimeBodyPart();
messageBodyPart.setText("This is a forwarding email\n\nThe original content is following:\n" );
Multipart multipart = new MimeMultipart();
multipart.addBodyPart(messageBodyPart);
messageBodyPart = new MimeBodyPart();
messageBodyPart.setDataHandler(message.getDataHandler());
multipart.addBodyPart(messageBodyPart);
forward.setContent(multipart);
Transport.send(forward);
System.out.println("Message has been already forwarded ....");
}
```

Figure 21. Function of forwarding an email

Figure 21 demonstrates the function of forwarding an email. The first messageBodyPart is created for storing the text came from the host address, the second messageBodyPart is created for the forwarded content, and the two messageBodyParts are combined into the multipart as a content to be sent.

6.3.5 Email connect with cell phone

The messages transported between email server and cell phone has become an innovative media to communicate with each other. The messages' transmission is not limited among mobile phones or email accounts. With the network development, an email message can be as a text message is sent to a mobile phone. Short Message Service (henceforth SMS) is the protocol used by mobile phones corporations to sent short messages. Short messages are considered as short emails are sent among mobile phones.

To send a text message from email server, the mobile SMS gateways have to be provided by mobile carriers. The SMS gateway is the link between the World Wide Web and the cell phones through which short text messages are sent. Basically the gateway is a website that permits one to send SMS messages from a mail agent to people who is only available with the cell phone that is serviced by the gateway. The gateway permits the communication of SMS messages to mobiles from another media source through using standardized communications protocols. (iBlogger 2010.)

Sending an SMS messages to mobile from an email server, the number and the carriers of the recipient need to be known before starting emailing messages to a mobile phone. There is one carrier named Alltel from US, the recipient address is written "mobile phone number@message.alltel.com". The following are also some other SMS gateways provided by carriers:

Airtel (Karnataka, India)	number@airtelkk.com
Fido (Canada)	number@fido.ca
Koodo Mobile (Canada)	number@msg.koodomobile.com
LMT (Latvia)	number@sms.lmt.lv
Meteor (Ireland)	number@sms.mymeteor.ie
Movicom (Argentina)	number@sms.movistar.net.ar
MTN (South Africa)	number@sms.co.za
MTS (Canada)	number@text.mtsmobility.com
T-Mobile (Austria)	number@sms.t-mobile.at
T-Mobile (UK)	number@t-mobile.uk.net
Telus Mobility (Canada)	number@msg.telus.com
Vodafone (Italy)	number@sms.vodafone.it. (Esengulov 2008.)

7 CONCLUSION AND DISCUSSION

This chapter includes the results of the thesis, limitations and challenges of the research project, further development and conclusions on my thesis work.

Email system has become an indispensable media in people’s daily lives. With the software engineering development, the services and functions of email systems are also being constantly improved to meet the expectations of the users. The user friendliness requirements are mainly for users with simple operation. The security and safety requirements have to be strengthened to protect the personal or business privacy information.

The architecture and functionality of the resulting system were designed by UML on the basis of the knowledge of analyzing the existing email systems. Furthermore, the Meta Object Facility is used as a standard to define the diagrams designed of architecture and functionality in chapter 6 of my thesis project.

The research work has its limitation, despite the fact that the research questions were answered during the study process. The compatibility needs to be promoted between the various email agent servers and the end-user applications, which brings more convenience for the email users. Therefore, it is necessary that the web mail agent servers are compatible with end-user email systems in further development.

To conclude my thesis work, I learned all knowledge about email systems and the JavaMail Application Programming Interface, which improves my scope of my experiences. Moreover, the time for this research project was tight. I also faced some problems were difficult to manage with my own knowledge. Additionally, designing complicated functions such as new email remainders is time consuming since it needs advanced complicated language programming skills, which is out of my research scope. I managed to achieve the objectives of my research work which will help do further research for the interested reader on this area in future.

REFERENCES

Printed

Bennett, Simon & Skelton, John & Lunn, Ken 2001. SCHAUM’S OUTLINE of UML. MacGraw-Hill International (UK) Limited.

Hair, Joseph F. & Money, Arthur H. & Samouel, Phillip & Page, Mike 2007. Research method for business. John Wiley & Sons Ltd, West Sussex, England.

Järvinen, Pertti 2001. On Research Methods. Opinpajan Kirja, Tampere, Finland.

Kotonya, G. & Sommerville, I. 1998. Non-funcitional Requirement. John Wiley & Sons Ltd, West Sussex, England

Sekaran, U 2003. Research Methods for Business: a skill building approach. John Wiley & Sons, cop.

Sommerville, Ian & Sawer, Pete 1997.Requirements Engineering. John Wiley & Sons Ltd, West Sussex, England.

Ryabov, Vladimir PhD 2010. Research methods, Part2. Kemi-Tornio University of Applied Sciences, Finland.

Not printed

Braun, David & Sivils, Jeff & Shapiro, Alex & Versteegh, Jerry 2001. Unified Modeling Language (UML) Tutorial. Read August 28, 2011.
< http://atlas.kennesaw.edu/~dbraun/csis4650/A&D/UML_tutorial/index.htm>

Borenstein, N & Freed, N. 1993. MIME-Mechanisms for Specifying and Describing the Format of Internet Message Bodies. Read August 13 2011.
<http://www.faqs.org/rfcs/rfc1521.html>

Cyberindian 2008. What is IMAP and what are its specific advantages over POP3. Read August 13 2011.
<http://www.cyberindian.com/web-hosting/article.php?article_id=88>

Esengulov, Aibek 2008. How To Send Email To Any Cell Phone (for free)
<http://www.makeuseof.com/tag/email-to-sms/>

iBlogger 2010. What is SMS Gateway? Read November 12, 2011.
<http://indianblogger.com/what-is-sms-gateway/>

Jones, Brad (jGuru) 2001. Fundamentals of the JavaMail API. Read July 30, 2011.
<http://java.sun.com/developer/onlineTraining/JavaMail/contents.html>

Malan, Ruth & Bredemeyer, Dana 2001. Functional Requirements and Use Cases. Read October 16, 2011.
<http://www.bredemeyer.com/pdf_files/functreq.pdf>

Maguire, Martin & Bevan, Nigel 2002. User requirements analysis-A review of supporting methods. Read October 7, 2011.
<http://www.nigelbevan.com/papers/WCC_UserRequirements.pdf

Nielsen, Jakob 2003. Introduction to Usability. Read October 28, 2011.
<http://www.useit.com/alertbox/20030825.html>

NetBean 2011. NetBeans IDE 7.0.1 Release Information. Read November 06, 2011.
<http://netbeans.org/community/releases/70/>

Nemil, Tony & Hemrajani, Anil 1999. Introduction to the JavaMail API. Read July 28, 2011.
<http://www.javaworld.com/jw-06-1999/jw-06-javamail.html?page=4>

ORACLE 2011. JavaMail API Introductions. Read August 14, 2011.
<http://www.oracle.com/technetwork/java/javamail/index.html>

OMG 2011. OMG's Unified Modeling Language and OMG's Meta Object Facility. Read October 18, 2011.
<http://www.omg.org/mof/>

Su Ran 2010. A research about email accounts growing. Read August 23, 2011.
<http://news.iresearch.cn/viewpoints/113175.shtml>

Snoke, Grace 1999. What is SMTP Authentication? Read July 30, 2011.
<http://www.ehow.com/facts_5862931_smtp-authentication_.html>

Taylor, Dena 2009. The Literature Review: A Few Tips on Conducting It. Read August 17, 2011.
<http://www.writing.utoronto.ca/advice/specific-types-of-writing/literature-review>

Trivedi, Yatri 2011. HTG Explains: How Does Email Work? Read August 04, 2011. <http://www.howtogeek.com/56002/htg-explains-how-does-email-work/>

Zukowski, John 2000. Fundamentals of the JavaMail API. Read August 18, 2011. <http://www.digilife.be/quickreferences/pt/fundamentals%20of%20the%20javamail%20api.pdf>