Development of Textual Email Client in Java

Xing Hu

BSc (hons) Computer Information System
May 2005

Development of Textual Email Client in Java

Submitted by Xing Hu

Copyright

Declaration

This dissertation is submitted to University of Bath in accordance with requirements of the degree of Bachelor of Science in Department of Computer Science. No portion of the work in dissertation has been submitted in support of an application for any more degree or qualification of this or any other university or institution of learning. Except where specifically acknowledged, it is the work of the author.

Signed.....

This thesis may be made available for consultation within the University Library and maybe photocopied or lent to other libraries for the purpose of consultation.

Singed.....

Abstract

Email client is one of the most successful computer applications, it has been widely used but as the technology develops, it is exposed some deficiencies and cannot satisfy users' need sufficiently nowadays.

Aiming for maximising the utility of the email client, in the way of enabling sorting messages functions and increasing efficiencies, my approach is to develop a textual email client which helps to build multiple search expressions and then to construct hierarchical virtual folders to save these queries.

I adopt java language to create my system and detailed methods will be presented in my thesis.

Acknowledgement

I would like to thank my project supervisor, Julian Padget for his great help throughout the project; also I would like to thank my parents for their support. Moreover, I would acknowledge my friends, Yang and Yan who helped me in proofreading.

Contents

Chapter 1

Introduction

1.1 What is Email Client?

Currently the most frequently used communication system is Internet; meanwhile, message communication is the one of its most favourable functions. Through the email transformation, user can reach the destination and link to every corner of the world at a very low lost. These messages can be in the form of text, image, audio and other contents. Moreover, user can get free news; there is no other traditional methods can be competed with.

Email client it is the program which user uses to work with email. Besides, it allows users to interact with email in a much friendly manner. It is a program for viewing, composing and storing emails e.g. Microsoft Outlook, Outlook Express, Mozilla Thunderbird. [1]Protocols supported by email clients include POP3[1] and IMAP[2] . IMAP and the updated IMAP4 are optimized for storage of email on the server, while the POP3 protocol generally assumes that the email is downloaded to the client. The SMTP[3] protocol is used by most email clients to send email.

Email Client, which has been developed accompanying the email, it has many functions than before. To assist users better, different approaches are still studied on it in various directions.

[1]Post Office Protocol version 3 permits workstations to dynamically access a mail drop on a server host (TCP/IP).

[2]Internet Message Access Protocol, it is a mail protocol that provides management of received messages on a remote server.

[3]Simple Mail Transfer Protocol is an internet protocol used to send emails between servers.

1.2 Existing Problem

Besides the great success of email client, there are still a variety of different elements threatening corporate email system and while some are widely known, such as email viruses.

The increasing popularity of email enables it to have become one of the most important communication media which conveys and transports information. Users might have to handle high volumes of messages retrieving and sending everyday. Reasonably, users now are more concerned about how to manage and organise their messages. [2] As can be been seen from a survey, the average number of filed messages is 858 compared to inbox items is 2482, they revealed the possible reasons are (a) people use the mailbox as the task manager which could remind them of current task, so they keep the information accessible in mailbox; (b) it might be hard to file information to remove it from the inbox because of filing process is difficult. Also, the survey has identified that, if some messages require a certain amount of time or effort to process, users would delay dealing with messages and leave them not discharged. Filing the message is cognitively difficult task [3],[4], successful filling will be highly dependent on being able to imagine future retrieval requirements.

To handle this problem, various approaches have been tried out, some are to build folder hierarchy to cope with big number of emails, but it will raise another problem of a more extensive use of the folder. Alternatively, a better way is to embed a search function in email client and this is what I am concerned in my project.

1.3 Overall Aim

With effort to manage emails better, the overall aim of this project is to produce the virtual folder and multiple queries using combination of logical operators with fundamental functions in the text-based email client. The email client should support users to achieve the design goal with efficient searching mechanism.

Chapter 2

Literature Review

2.1 Introduction

Email emerged in last century which just has a short history but with robust functionalities, it can be said that, it's another innovation in the human contact method.

Email Client, according to its definition, it is an application which can be run in user's personal computer to enable user to send, retrieve and organise the emails. The email systems are based on a client-server architecture, and message is sent from many clients to a central server, which re-routes the mail to its intended destination.

The project I am working at is textual email client which will be mainly designed and implemented by Javamail API. Actually there are many different email tools designed in a variety of forms with different features to attract the users to utilize their communication methods. Graphics email clients with vivid and friendly interface; it is popular used nowadays. Unlike to the graphics email clients, plain text email is just that - in the format of plain text. They are nothing fancy, there are no pictures embedded in the email, and there are no special fonts. Plain text emails are simple. The term plain text refers to textual data in ASCII format. The reason why I choose to do it in text-base is because plain text (also called clear text) is the most portable format because it is supported by nearly every email application on various types of machines. Besides, without too many graphics to download, it will be much quicker to perform the email management. I will separate my report in different chapters in order to make my idea more clearly.

2.2 Background

2.2.1 E-mail Client's Advantages

Email, together with Telnet and FTP, gave the internet its momentum. Since the 1970s, email has evolved into the communication tool of the choice for information technology academics and professionals.[5]Email now is the a very powerful and efficient contact tool, people use it to send letters, notes, files, data or reports to their customers, friends, colleagues and families. They don't have to worry about the physical distance between their communication objects and there is no time restriction.

Basically, there are four types of emails clients which are:

- Linux and Unix Email clients
- Mac Email Clients
- Web-Based Email Clients
- Windows Email Clients

To manage the emails, email client is aimed to help people to manipulate better with the emails. Different emails clients have different features; I will list some of their advantages:

- Email Client can help people to manage their messages quickly. Suppose you are away from your personal computer but still have to deal with emails which give you updated information, the easy is to use a web-based email client or download the email client to use. Web-Based Email Clients which can be used anywhere from any computer to use, people dont have to worry about the assets which is limited in the computer itself.
- Windows Email Clients, for example, Outlook Express[6] offers that if you share your computer with other people, since it can be configured for use by a number of people, accessing a number of email accounts.

2.2.2 E-mail Client's Problems

- Some email systems can send or receive text files only. Even though the user can send and receive images, programs, files produced by word processing programs, or multimedia messages, some user may not be able to properly view his message due to lack of supporting functions.

- Hard to identify the important messages in user interface. Users are aware of emails' coming which they are concern with. But currently email client can not provide the function which can highlight the significant ones with another colour or symbols. People are expecting about automatic filling schemes intended to help bring the orders of chaos.

- Can receive too much or unwanted email. User can receive "junk" email which costs extra time to deal with. On the Internet, junk mail is called spam.[7] It might have viruses and offensive language which it is time-consuming and under risk for users to deal with them.[7] User may have to take active steps to delete the email user receive and try to stop it from being sent to user in the first place.

- User can not separate the email with the attachment. [8] If user want to delete the attachment which occupies too much space but keep the original email. Currently no email client has this functionality, so user has to act the deletion for both attachment and email, or save the email in somewhere else first, then delete it with attachment.

- Lack of context support. Emails provides very limited context for new messages in longer and more complex interactions. [9] For example, the same author has sent two messages; they might have the same topic, or might be part of a conversational thread. User needs more dynamic way to connect the related messages and to increase context.

- User can not perform the multiple searches within the emails. Since Gmail and Thunderbird has been released the search function, it becomes possible that people can do the search function and store the result. But, it still has problem that now there is just single search at each time which will result in tedious repeats.

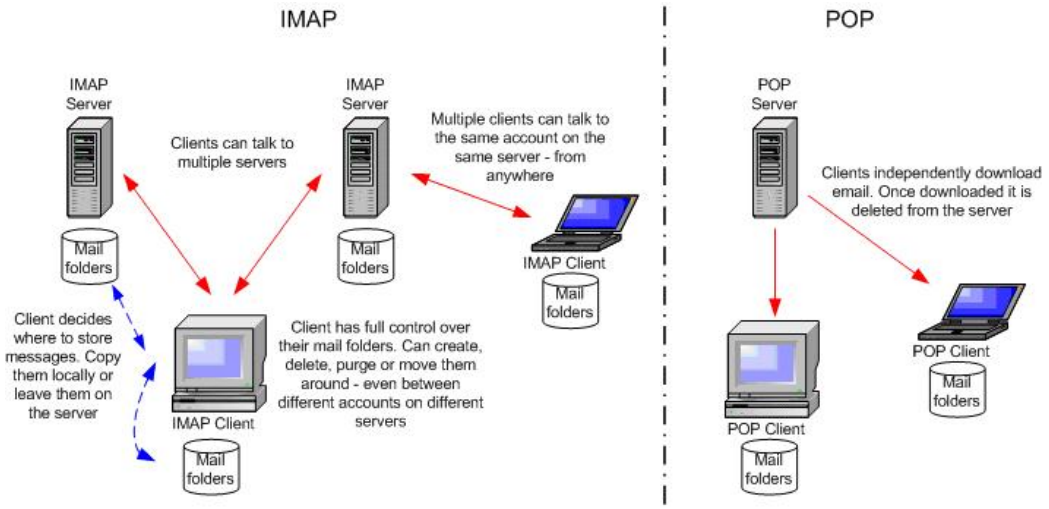

Figure 2.1: imap and pop

2.3 Relevant Information

Before launching an email client, user should have some information from the Internet Service Provider (ISP) so that user can configure the client properly. The following lists a few important things user may need to know:

2.3.1 User Email Address

The email address user will use to send and receive mail. This is usually in the form of userrname@userrisp.net.

2.3.2 Server Type for Receiving Email (POP or IMAP)

In order to receive mail, user must know what type of server user network administrator or ISP is using. This POP or IMAP address, is usually in the form of mail.someisp.net. I will carry out my work using IMAP, so now I will generally compare POP and IMAP to support why I choose the first one, see figure 2.1.

2.3.3 POP Sever

POP, which stands for Post Office Protocol, is designed for "offline" mail processing.[9] It is used to send email from a mail server(usually shared) to user's email client's inbox, the place where incoming email is stored. Generally speaking, user interacts with sever to download and save their emails. Once the email is delivered to user's computer, the message will be deleted from the sever. POP emails just can reach a single destination, and it is not well-suited for the goal of accessing user's inbox of recent messages or saved-message folders from different machines at different times.

2.3.4 IMAP Sever

IMAP, which is short for Internet Message Access Protocol, is a protocol for retrieving email messages from user's ISP's(Internet Service Provider) email server. IMAP [10] supports offline email processing, but still, it has special functionality which is in online and disconnected operation. Differs from POP in that email from IMAP servers are stored on the server and stays there even as user download and read user mail, whereas POP mail is downloaded to user's email client directly and will no longer exists on the server.

On the contrary, remote client reads the messages from the server using IMAP email software. The messages are not all downloaded to the client PC. The client can ask for specific messages or search the server for messages meeting certain criteria. The user also has the option to save the messages locally.

The big advantage of the IMAP protocol is for the mobile user. The user just has to find a PC connected to the Internet to read their messages. The messages and folders are the same from every PC IMAP protocol advantages are:

- Can manipulate persistent message status flags.
- Can store messages as well as fetch them.
- Can access and manage multiple mailboxes.
- Can support concurrent updates and access to shared mailboxes.
- Suitable for accessing non-email data; e.g., NetNews, documents.

- Can also use offline paradigm, for minimum connect time and disk use.
- Companion protocol defined for user configuration management (IMSP).
- Constructs to permit online performance optimization, especially over low-speed links.

2.3.5 Server type for sending email (SMTP)

The Simple Mail Transfer Protocol (SMTP) is a protocol for sending email messages between servers. Most email systems that send mail over the Internet use SMTP to send messages from one server to another; the messages can then be retrieved with an email client using either POP or IMAP. SMTP is also used to send messages from a mail client to a mail server. This is why user needs to specify both the POP or IMAP server and the SMTP server when user configures his own email application.

2.4 Evaluations

2.4.1 My Design Goal

The main aim of my project is to establish the function of "virtual folder" and store the queries in different forms of searches. Nowadays there are many email clients which assist people to manage their daily tasks and remind them important events, etc. But according to fact that, there is a huge number of "heavy users", they have to deal with various emails from head offices, colleagues, subordinates, customers, friends and so on. So it is essential to them to classify their different types of emails in order to use them conveniently in future.

But the main problem is, most of these email clients don't support the search function to users when they are looking for a particular email which will be time-consuming for them to take the action. For example, a user who wants to find an email back to half a year ago which is stored in the inbox which might contain thousands of emails. Now he has to go through all the emails around the time and find out the each one's content to see whether he tackles the right one. This work is very troublesome to the users who have many emails to deal with but they don't have much time to organise them like creating especial folder to different coming emails.

On the another hand, set up too many folders which also will become the problems, for example, some emails exist in both inbox and folder which take extra space of the mailbox, or, people forget which folder they put the emails in, they have to do the research again.

So, to help the people out of the obsession situation is the main purpose of why I will design this email client. The basic sketch is to build the function that can support to carry out the mission. If I want to search the pattern like, use the logic operator, AND, OR, NOT to carry out the search not only for single value but with multiple values.

2.4.2 Evaluate Some Email Clients in This Area

To get ideas and enumerate their own advantages and flaws to emphases the importance of designing queries, I will discuss different types of email clients as following:

2.4.3 Frequent Use Email Client

Outlook, which is established by Microsoft, which is best suited to larger organisations with IT support for the email sever side, it keeps increasing its popularity.[11] It allows different users to create multiple email accounts, and provides some brilliant problem resources such as marking things as important or outstanding, managing deadlines and reminders and so on. But, according to the analysis [12], it reveals that users are not making use of many of these resources, it suggests the reason may be due to it is hard to discover and use them.

2.4.4 Current Email Clients with Virtual Folder

Firstly, I use two examples to give details of what the virtual folder is and my comprehension about it.

The first one is the Gmail which is launched by google has this function [13] which uses the SQL queries to create and maintain "virtual folder". The idea is, instead of individual folders containing all the messages of each email list or subject, or sender, or what criteria are used to define the folders, Gmail creates virtual folders which are simply SQL "views" of the message database. From the Gmail's User Guide, it addresses that, "Its main feature is the powerful vfolder mechanism. Imagine all senders' emails are sitting in the one big folder, but user can look at it in different ways (queries). These queries happen at run-time. So user can have a virtual folder (vfolder) which gives user all messages with 'ham' in the subject. And user can have another vfolder which gives user all messages with 'pork' in the subject. And if someone sends user a message with 'pork and ham' in the subject, it will appear under both queries!"[13]

From its instructions we can see, it's not tend to create the new folders but to set up some virtual files which can group the emails in the search result, further more, the emails are not copied to the new position or somewhere else, they are exported when applying the MySQL tables, user still can delete, or mark them is both root folder and virtual folder. Each action will affect the email existence.

Another example is Thunderbird. [14]Beginning with version 0.9, Thunderbird now allows user to create Saved Search folders. A Saved Search folder looks like a regular mail folder but when user clicks on it, it runs a search according to criteria that user sets previously and it displays a list of messages that match those criteria. For example, user could create a Saved Search

folder that lists all the messages received from a certain person over the past 30 days, even if those messages are stored in different folders and subfolders. A Saved Search folder is a "virtual folder" in the sense that it merely displays a set of messages that meet the search criteria, while the actual messages remain stored elsewhere. If user selects and deletes a message inside a Saved Search folder it will get deleted from its actual location, but if user delete a Saved Search folder itself all of the actual messages will remain intact. Moreover, unlike a normal folder, if user modifies the search criteria for a Saved Search folder its virtual "contents" will be accordingly updated. User Saved Search folders will remain in the folders pane even after user exit and restart Thunderbird, until user delete them, thus giving user quick and convenient access to user pre-defined searches.

2.4.5 Textual Email Clients

These two are all graphics email clients; there are also some text-based clients, for example, Pine. Pine is launched by University of Washington, it's menu-driven, easy to use, includes an address book and mail "folders," supports MIME(A communications protocol that allows for the transmission of data in many forms, such as audio, binary, or video) attachments, and even comes with its own easy to use text editor called pico. Pine calls pico whenever user edits mail, but user can also run pico as a separate stand-alone text editor. Pine software is much faster than web-based email clients since it has no graphics to deal with, however, attachment processing can be a bit cumbersome.

2.5 Design

2.5.1 The language I use

I use Javamail API to be my design language because it is platform independent, it can work on Windows/Unix/Mac, which means the email client will be portable and can be participated in different environments.

Secondly, for Javamail API, it has classes which can help to design the email client, which means I don't have to design the query by other methods to retrieve and send the emails.

2.5.2 What kind of search I want to perform

The interface of my design is textual, so it will be adopted command lines to carry out the tasks. The essential function, the search field will be distinguished in these categories:

- Name
- Subjects
- From
- To
- CC
- Key Words

The more important design is about the multiple searches, for example, we may carry out a search as "sender contains= colleagues AND friends", "friends = John NOT Martin", since javamail API provides the methods to achieve the goal; the main packages are [15]:

javax.mail	Classes modelling a mail system.
javax.mail.event	Listeners and events for the JavaMail API
javax.mail.internet	Classes specific to Internet mail systems
javax.mail.search	Message search terms for the JavaMail API
com.sun.mail.imap	An IMAP protocol provider for the JavaMail API that provides access to an IMAP message store

It supports these three logic operators: AND, OR NOT. Furthermore, search should be carried out in the term of all the messages and different times.

2.5.3 Further thinking

Since these searches will be designed and implemented within Javamail API, as shown before, Gmail uses the query which is used MySQL technology providing robust search capability. This is another inspiration in the future email client design.

2.6 Summary

This project is to establish an email client which has "Virtual Folder" to support the multiple searches within the emails. So far, it has been shown that its interface will be textual and the design language will be Javamail API. From this literature survey, I have learned more about some background information about POP, IMAP and why I choose the later one. The design will be carried out based on these principles; further questions will be considered in real programming.

Chapter 3

Requirements and Specification

3.1 Sources of Requirements

This chapter will discuss the relevant requirements in my email client, and these requirements will be transformed to my system functions.

To get the requirements, the user, the technology and the evaluation existing system should all be integrated to fully cover the whole design process.

3.1.1 User Identification

Users are the main influences of designing the system; no user, which means no market, therefore there is no value to use. Now email client has been used heavily in companies, hospitals, industries, universities, etc, so it is deigned towards the frequent users among these institutions. With special feature of searching, it will provide more convenient function to use. As learnt from the analysis [16], these users check their emails everyday; somehow, email checking is as a routine or task management for supporting their works. So this email client has much potential space in competing with other email clients to make user's life much easier.

Different user groups are managers, students and teachers, businessmen, engineers, system designers and so on. They are almost all the busy email users and have email using experience. To get appropriate feedback from them, I interviewed couple of students who are from university to reveal what is the focus for the design conversation, showing how the work hangs together rather than breaking it up in lists, identifying what matters in the work and guiding the system structure from the proper feedback.

I assume they can represent what most users want since they have to deal with huge messages each day, therefore, user requirements got from here will be seen from the subsequent user requirement section.

3.1.2 Design Language Requirements

Designing a textual email client it needs the flexible and adoptable programming language to use via different platforms. It will be beneficial as the independent language,once the operating system is changed; however, system will not be affected. I will perform the actual design in Windows platform according to my operating environment, but other different platforms should be considered to enlarge its usability according to individual user preference. Hence, the language also should be portable and dynamic.

Another issue is, design language should be easily to implemented and maintained, for single language selection to use in system, it should be easily understood and widely accepted.

3.1.3 Evaluation of Existing Systems

In the literature review, I classify three kinds of different email client which are

1. frequent use email client
2. current email client with virtual folder
3. textual email client

From these evaluations, we can summarize as:

1. Email clients can perform in various ways not only as email dealing system, for example, helps in managing user's schedule, but user finds this function is hard to discover and use;
2. Multiple design languages might be used in an email client;
3. Different formats of virtual folder will be created with different messages manipulation, for example, in gmail, if user chooses to delete the message in virtual folder, it will affect the messages in root folder, but by contrast, message still exist in Thunderbird email client version when user choose to delete virtual folder completely;

4. Textual email client can be run faster than normal web-based email client even it has problem in dealing with attachments.

It gives good hints in designing my textual email client; I decide the criteria would be

1. My email client will focus on the virtual folder and multiple search function;
2. Carefully selecting the design language and consider the operating environment;
3. The attachment dealing will not be considered in my textual email client design, as it requires a high-level security support.

3.2 Functional Requirements

These requirements are generated from the literature review at the initial design stage, email client as a operating platform, it has the various requirement which will be shown in following sections.

3.2.1 System Requirements

Retrieve the messages

Once user logs in the textual email client, system will ask him to input the hostname (which requires the imap sever name), username and password to access to the system.

User then can get the messages from the server, and system will display the message where the message is from, the sent time, the flags which indicate whether the message has been seen or not, the subject of message as well as message content.

Send the messages

User then has the option which is to send the messages. The system will require the server name(which is smtp address), the user name, the password accessing to the account, the account which user uses to send the messages, the address to which user sends messages, other addresses which user wants to send together, the subject of messages, the content of messages user constructs.

Create the messages

To send the message, user should specify the receiver address and compose the email content first. Email client should offer the all the message components to user although as the fact of some users do not have the habit of entitling the message.

Save the messages

User is able to save the message in client or export out to local machine. According to two types of server, pop and imap, either message will be downloaded in user's computer or be manipulated in the server. Further more, if user has not finished a message, it can be kept in the draft folder and user can construct it later on.

Forward the messages

Message can be sent by original user to other recipients otherwise it will take extra time for users to copy and paste.

Delete the messages

User has the authority to delete unwanted messages in email client.

Create the folder

User would like to create his own folders in system in order to categorize the different messages.

Rename the folder

If user feels like changing the name of folder name, they can enter another folder name as they prefer.System will display the new name of folder.

Delete the folder

User is able to delete the folder to organize the email client, but in the security reason, the default folders such as INBOX should not be deleted.

Save the message address of sender

Email client provides the function as to save the sender's address in memory, user will not be bothered to memorize too much information and he just needs to open the address book to insert the demanding message address to the message header.

Classify the messages in different folder

User can organise his emails and place them into different folders. Once user has logged in the system, system would sort the new incoming messages into different categories by their importance flags.

Send the attachment

Email not only can convey the information by itself but also has a feature which is to attach a file, a piece of audio or a picture to destination address without constructing in the message content or in another message.

Prevent Spam Messages

It's annoying to most users to receive the advertising messages or from the unwanted person. It's necessary to tell the email client to place these messages into junk mail in order to prevent harassment. Email client will memorize the specified email addresses and prevent them next time.

Search the messages

Multiple search for particular message. It's an interesting part of my email client design. Single query can be time-consuming and it has to be done several times to find the exact email. Thus, it demands high level search which is to build the multiple searches covering the whole section of the email. There are several areas user wants to do the search as:

1. The header section, which has the email attributes such as "from", "subject" and so on, user might want to do the queries among these values.

2. The body part, the content of message it might contain some information the user is concern about, in order to find out the message, search function must consider this part.

3. On the topic of search algorithm, the logic operation can be applied in multiple fields. The operators are "AND", "OR", "NOT" which specify the searches, in other words, the query expression will include and exclude the key words to narrow the region of messages.

4. The query can be performed in the same sender to get all the messages, or the messages containing the same words from the different senders.

5. In order to create the easy use query, the search will not use mathematic symbols or strict structures. Users will be given the example in the text field to do the search; they have not to memorize academic forms.

Save search result

To retain the result, creating a folder which is under the root to keep the searched messages is essential.The search folder is at the base of "INBOX", all the messages are stored in the virtual folder can be viewed.

To save the search result is another improvement of email client; it builds the virtual folder in the system and stores messages for user to take further actions. User does not have to look through the whole folder again,instead,they just need to check in the stored result.

Mark the message

User can score the message as level of significance to remind himself. Message once be marked can be listed in the level of importance. The mark can be removed or changed to another level.

Manipulate the message and attachment independently

User sometimes finds it's hard to work on the message and attachment separately. For example, the attachment is too big and takes much space of email client, but content of message is important to keep. Now user has the only option is to export the message to computer and delete the message completely. To find the method to separate them is interesting study direction of email client development.

Make email client as task management assistant and reminder

Since more and more companies and individuals rely on email to plan the daily tasks, email client, somehow, can add the function to deal with users' tasks and have alarm to remind the important event on certain day.

Message filtering

We are used to hear the news of viruses within the email or attachment, although user has noticed its severe damage, it is still impossible to defend effectively by its disguise and fast spread speed. It is important to filter the messages by scanning the viruses. The scanning process can be performed automatically by the email client and it gives the warning once the virus is found.

Data type support

Message contents may vary in different data such as various languages, character fonts, size, etc. User may have the experience of getting the unreadable message, one of the reasons is the email client does not support the data type.Email client would be designed to read different formats.

Support HTML/XHTML/CSS content HTML

HTML email newsletters are vivid conveying picture information to readers. Users would get Web pages which is much easier to view and navigate than a top-down plain text email. But the email client may be not able to retrieve

the graphics and it may contain offensive contents to users. To make the content visible, email client needs to code the HTML email for user's reading.

Signature

It's that fun tag at the end of an email message that user can use to provide contact information, or share user's personality. It's a special and unique symbol which user creates for himself only. To make the email client friendlier, personal and interactive, allowing user to make signature is an attractive feature in email client.

3.2.2 User Requirements

User requirements, which are part of the functional requirements, they show user's understanding of the system without detailed technical knowledge. From the previous interviews, user feedbacks are collected and they are generated and organised to form the user requirement. So in this section the requirements are:

1. User will not be required high level of computer knowledge, in other words, users will not be asked for academic background to use textual email client. It has the same functionalities as some other system by logging in , inputting the username and password so user will not feel unfamiliar to use. Once user makes the mistake in operating, system will give the error message to remind user to try again.

2. Easy understanding and fast corresponding interface is demanded by users, they declare that they do not like the delaying of waiting to download the system;

3. System would perform all the tasks correctly and will not cause confusion in using;

4. System can give instructions in wrong operations;

5. System can recognise the emails in time order, with new or other flags to classify them into different groups;

6. System connects to server timely as user specifies;

7. System automatically reply to senders when user sets status as busy;

8. System will remind users when new incoming messages arrive;

3.3 Non-functional Requirements

The non-functional requirements, [17] which are not directly concerned with specific functions delivered from the system, they will be concerned about the operating environment, reliability, response time and store occupancy. Still, they might define the constrains of I/O device capabilities or data representations.

Related to the design, the non-functional requirements will be

Usability requirements

System should provide a friendly interface and easy command operations.

Efficiency requirements

Systems will response quickly to user input and display the output.

Reliability requirements

System will be consistent to deal with unwanted situation to maintain information response in proper behaviour.

Space requirement

It will not take much space of memory and it does not have to install.

Portability requirements

System will be flexible to use in any platform since java design language is adopted.

Implementation requirements

System will be easy to implement in java language of a clear program structure.

Delivery requirements

System will be finished by 16th, May, 2005 to delivery.

Privacy requirements

System will be protect user's data and will not cause conflicts with current legislation.

Safety requirements

System is designed to meet all the technological requirements to minimize the loss of user's data.

Ethical requirements

All the information used on the system should be permitted from the authors and references of data sources in order not to break copyright law.

3.4 Functional Requirements Evaluation

Requirements are from various channels of sources, it is worth to re-think and keep the relevant requirements to be the direction of design. To find out appropriate requirements it needs to evaluate the original requirements.

Requirements should be checked its validation for realism, consistency and completeness. In this process, unrealistic requirements and errors will be discussed. The checking process will centralise the requirements from all users and compromise the community. The requirements should not cause the conflicts and it should be complete to include all the system functions. Based on currently knowledge and actual condition, realism checking will ensure requirements can meet actual design and implementation.

From my literature review, I have addressed the design goal is to build a textual email client which will focus on searching the folders for messages and save the searched result in the folder hierarchy. According to objective, the requirements will be justified in case the main purpose of design will not be diverted.

To verify all the requirements, I make a table to compare requirements and system design constrains, all the details can be seen from Appendix One.

After evaluation, system's requirements are specified, to view whole picture of system, the design functions of the systems are listed:

1. To receive the Internet messages, this involves building the message, connecting to receiving server and displays the messages in folder;

2. To send the message to multiple addresses, involving establishing the connection to sending server, creating the message address and content;

3. To search the messages in the default folder, involving connecting to the server, specifying the single and multiple searching queries and sorting the corresponding messages to virtual folder;

3.5 Summary

To sum up, the functional and non-functional requirements are identified in this chapter, and evaluated results will be better suit the system design goals.The design process then can be pursued based on requirement evaluation. But one thing needs to be keep in mind that is requirements oriented from the user and system are changeable, managing requirements should be in the total system design. Managing methods are requirements analysis and specification adjustment along with implementation modification process.

Chapter 4

Design

4.1 Introduction

My system is the textual email client, so which means the interface is text-based and is controlled by the command lines. Setting the components in the console screen is intuitive for users to view, unlike the graphic interface; user does not have to attain high level of GUI knowledge and just following the instructions on screen to input the commands. As an analysis also stated that, [18] more than thirty years after the advent of email systems, most users still view a simple list of message in chronological order and select from among those messages which ones to view in more details. All this indicates the feasibility of using textual email clients.

To analyse the system, I will give the details of possible solution to achieve the design aim and which will be aligned to evaluated requirements.

4.2 Architecture of the System

System has various entities and methods; they are linked and grouped to achieve the design goals. Here are some models cited from theory and built to suit my system.

4.2.1 Client-Server Model

There are three aspects in the system architectural design; firstly, to meet the system functional and non-functional requirements, decision of system

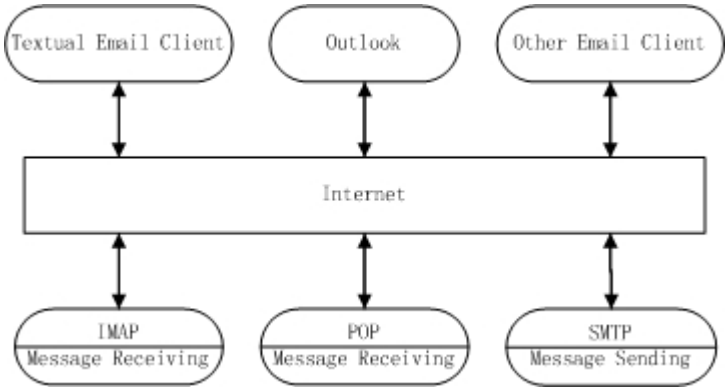

Figure 4.1: system architecture

organisation is important to build. There are a number of models to be selected from, according to my design, the client-server model [19] is more appropriate to use, because the system is organised as a set of services, by accessing associated servers. It has the following components:

1. A set of servers that offer services to other sub-systems. As I identified in the literature review, for the incoming messages, system will connect to imap server and smtp server for outgoing messages;

2. A set of clients that call on the services offered by servers.For example, my email client and outlook both can connect to servers to receive or send messages, although for incoming messages, outlook uses POP server, on the contrast, my email client calls on imap server of university;

3. A network that allows the clients to access these servers. I am able to establish the connection by using the university network.

To connect to the servers, email client should know the name of servers and what services they provide. For my design and implementation, I will use the imap server of university which is imaphost.bath.ac.uk to collect messages and smtp server which is mailhost.bath.ac.uk to post. The following diagram will show my system how is structured by using the model (figure 4.1):

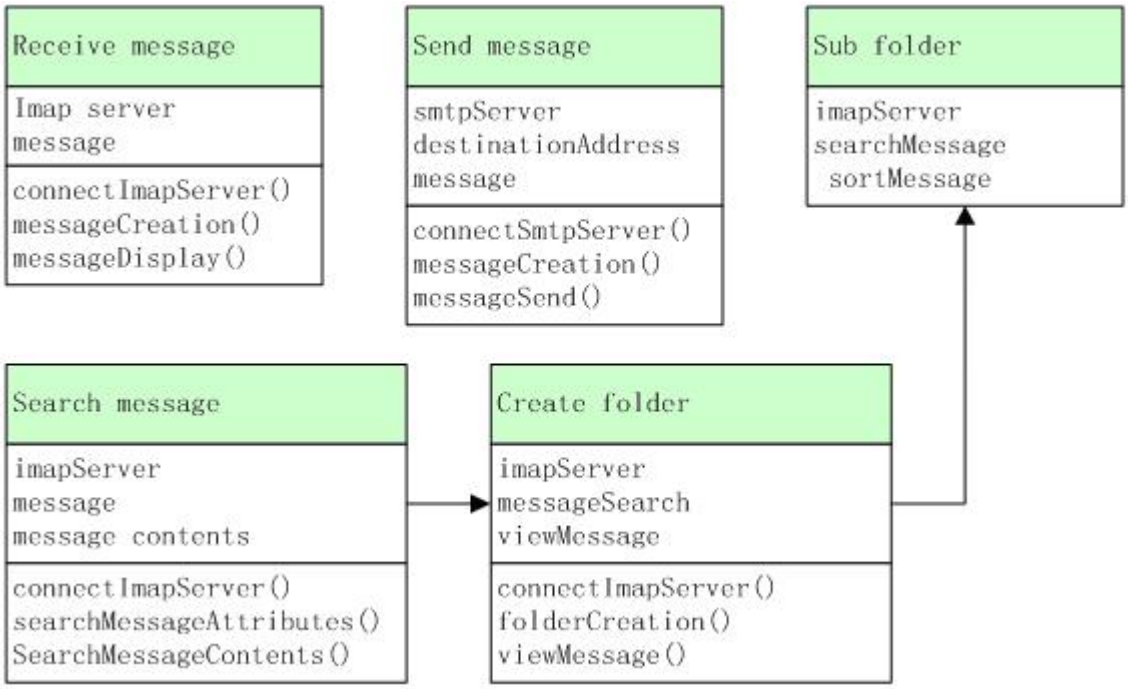

Figure 4.2: Class diagram

4.2.2 Modular decomposition styles

After applying the proper system organisation, the next approach is to decompose the system into modules. A module is a system component that provides one or more services to other modules. It may use the services provided by other modules. I will adopt Object-oriented decomposition to decompose the system to a set of communicating sub systems [19]

In my system, objects can call on the services from other objects. The benefit of using this model is when implementing the object; change one object will not affect the other objects. The modules are shown as below.4.2.

4.2.3 Task Flow

To show the task performance and different functions, each task flow will be produced in order to show them clearly.

see figure 4.34.44.54.6

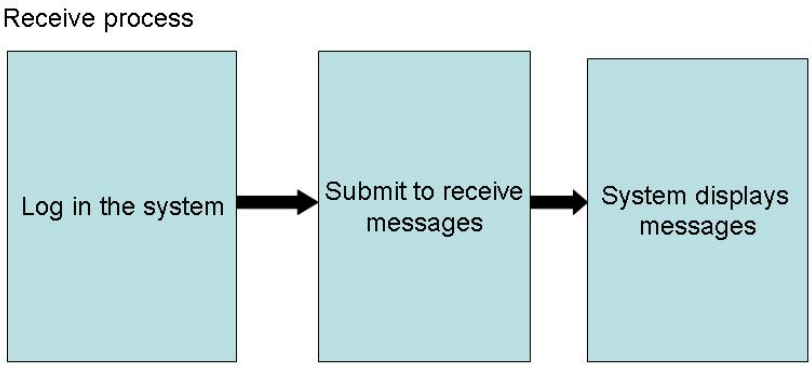

Figure 4.3: Receive process

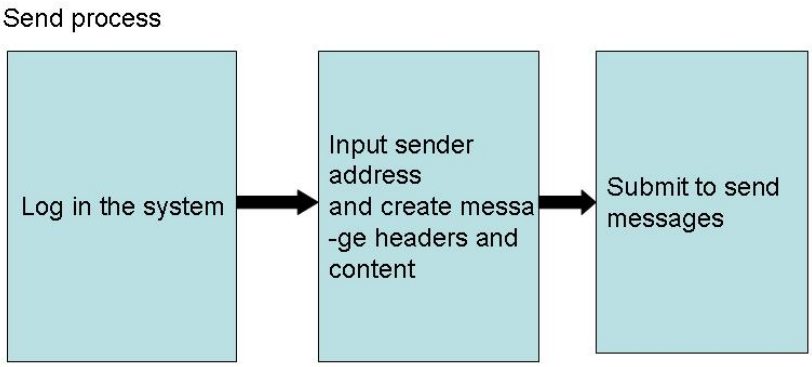

Figure 4.4: Send process

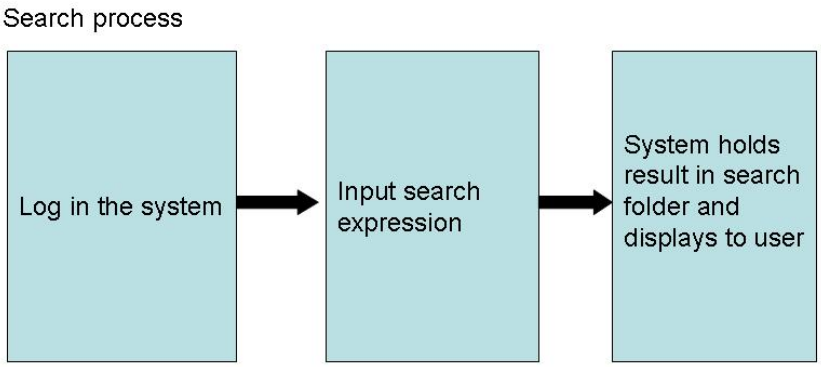

Figure 4.5: Search process

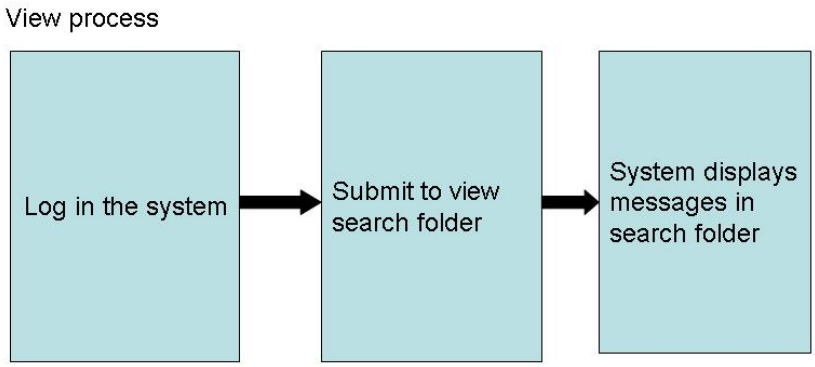

Figure 4.6: View process

4.2.4 System Structure

This demonstrates the system each module how to linked together. Form the start; user can perform on each sub system to do different tasks. This will give good hints on interface design as well.see figure 4.7

4.3 Design Criteria

As being discussed in the literature review, my design is to solve the problem of multiple searching and folder hierarchy. To show them clearly in design, I will explain in details in the following sections.

4.3.1 Functional Design Criteria

The fundamental functions of textual email client are the message retrieving and sending. The criteria schemes are:

1. Incoming server connection;
2. Message Creation;
3. System displays messages;
4. Outgoing server connection;
5. Message Creation with specific destination address;
6. System sends the message with message displayed;

4.3.2 High-Level Design Criteria

Conversely, higher level design specifications are explained in the multiple searching and virtual folder design assumptions. The possible algorithm and solution also will be shown in the following sections.

Multiple Search Design Criteria

1. Logical Operations

   User will type in these operators in console screen:

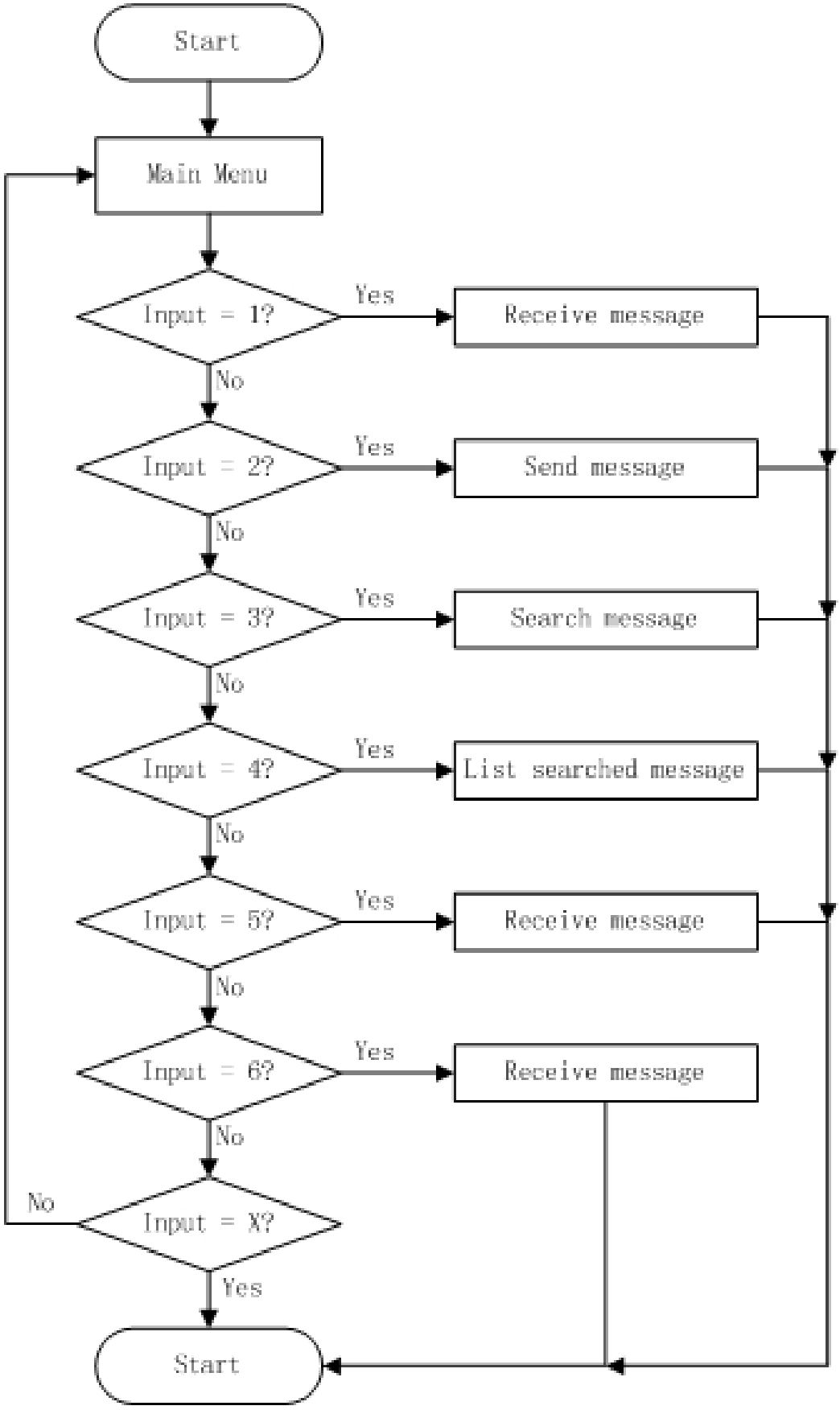

Figure 4.7: Structure

- AND-conjunction with two objects
- OR-selection among the objects
- NOT-disjunction of two objects

2. Brackets

   [ ]-use them to distinguish in the searching expression

3. Numbers and Characters

[0-9]	numbers
[abc]	simple characters
[NOT abc]	negation of abc characters
[AND abc]	union with abc characters
[OR abc]	intersection of abc characters
[a-z NOT fg]	a to z except fg characters
[a-m AND p-v]	a to z with p to v characters
[a-z OR ef]	ef characters
[a-m AND n-p NOT jk]	a to m and n to p except jk characters

4. Searching Items

   Here are the searching terms in the entire message, including the header attributes and body content. The items are:

   From

   It contains characters and search in the "from" area which has the sender's name. It is to search for text characters.

   Subject

   User can search both characters and numbers in the attribute of "subject", all the relevant messages which containing the specified queries will be displayed.

   Flag

   The flag's expressions are:

ANSWERED or UNANSWERED	messages has been replied or not
SEEN or UNSEEN	messages has been read or not
RECENT or OLD	show the new message
FLAGGED or UNFLAGGED	mark the message

   String

   1)Symbols

   These contains the characters which will be treated as in string field

"{", "}", "*", "@", "&", "$", "+", ";", ".", "^"

2)Email address Searching the email address from sender or cc group will be regarded as in the string.

For example: String=@hotmail.com [NOT @yahoo.com]

3) Email content String searches cover the whole message content.

One thing should be pointed out is that, in the textual email client, date has many formats and for a complete date expression is long, for example, if I enter in the wrong expression "3000-14-12", it's hard to correct. Besides, date should be checked whether it's valid before to show, otherwise it leads to problems, so in my email client, date searching is neglected for these reasons.

5. Single Search

   From=abc [AND x] [OR y] [NOT z]
   Name=abc [AND x] [OR y] [NOT z],
   Subject=abc [AND x] [OR y] [NOT z]
   Flag=ANSWERED [AND FLAGGED] [OR SEEN] [NOT DELETED]
   String=abc [AND x] [OR y] [NOT z]

6. Multiple Searches

   From=abcd [OR xyz], Subject=HCI [AND group meeting], String=course work deadline;

Virtual Folder Design Criteria

In my system, folder is an abstract class to contain the messages; what is more, it can still provide a tree hierarchy which is rooted in the default folder. To retrieve the messages, "INBOX" is reserved in the store which means "primary folder for user on this server".

Therefore, before setting up virtual folder, I assume the default root folder is "INBOX" which may contain certain amount of messages. Search will be operated in all the receiving messages since system will not store sending messages. Then, System will be designed to connect to the incoming message server with user name and password authentication.

When performing search methods, system will "scan" the whole INBOX messages to find the matching messages. The messages which meet the queries

will be copied and returned to the "Search" folder which is rooted from the "INBOX".

A Folder is initially in the closed state in case to protect all the messages that will be kept safely. The Message objects in the Folder are cached by the Folder. Thus, invoking messages in folder on the same message number multiple times will return the same message object, until an expunge is done on this Folder. However, I will not authorize users to delete and expunge messages in INBOX for security reason.

To avoid the folder overloading problem, I will specify the system to create only one search folder to contain the search result. Each time user carries out the search function; system will create a folder which contains the result and refresh it with new results. Since message numbers will be changed in INBOX, new search result will be modified accordingly.

The term of "virtual" which means the folder will exist in the folder group, all the messages from the server will copy to the folder instead of downloading to the local computer. Manipulate messages within the virtual folder will not affect the messages in the INBOX,it will prevent users from deleting important messages accidentally.

It is worth mentioning that, whether folder implementations will allow both messages and other folders in the same folder, it needs to be considered carefully in the language selection.

4.4 Email Client Prototype

Therefore, an email client prototype is to set up on these performances:

1. Prototype is able to connect to incoming message server to receive the messages;

2. Prototype is able to display messages in console screen;

3. Message header and body will be shown;

4. Prototype is able to connect to outgoing message server to send the messages;

5. Prototype is able to let users to construct the message;

6. Prototype is able to deliver the message to multiple receivers in "to" and "cc" group;

7. Prototype is able to search message in default folder;

8. Prototype is able to display the corresponding messages in console screen;

9. Prototype is able to save the corresponding messages in virtual folder;

10. Prototype is able to let user to perform single or multiple queries;

11. User can perform another query and prototype is able to refresh the search folder with new results;

12. Search terms are "Name", "From", "Subject", "Flag" and "String";

13 Combinations of queries using logical operators are "AND", "OR", "NOT" and brackets.

4.5 Summary

This chapter introduces the design approaches which are referred from the requirements from previous chapter. With model applied, it apparently exhibits the whole system structure and how the functions will be related to each other. The process of using the system also be displayed, further more, I go with the design criteria which are functional criteria along with two key parts, multiple search design criteria and virtual folder design criteria.

Chapter 5

Detail Design with Implementation

5.1 Introduction

This chapter is to present system design details. Besides, it will talk the modifications during the design process and stress the reason why I made these changes.

5.2 Design Delivery and Decisions

This dissertation, with appendices as well as program code will all be delivered to University of Bath, Computer Science Department on 16th, May, 2005.

Before going for the details, I will explain what language and other tool I use and why I choose to use them.

1. Javamail API

   Since there are many programming languages to choose from to design the email client, javamail API which is released by Sun Company has the robust functionalities which covers the need of designing textual email client. [20]The JavaMail architecture has been developed by JavaSoft and is part of the standard Java 2 API. It is available as a separate package and can be used from JDK 1.1.X applications. Using the JavaMail API offers the following architectural advantages:

– Well defined, abstract, object-oriented API for accessing messages via POP3, SMTP and IMAP. A single API can be used to access internal mailboxes via IMAP and external mailboxes via POP3. The JavaMail API hides the complexity of the underlying protocols by offering an object-oriented API to folders and messages.

– Plug-In architecture can be extended to support additional transport protocols.

– The source code of the JavaMail API is available for licensing by JavaSoft.

2. JAF(JavaBeans Activation Framework)

   [21] The JavaMail API leverages the capabilities for dealing with complex data types from the Java Activation Framework (JAF), which is part of the Glasgow JavaBeans specification. JAF provides Java with similar capabilities that plug-ins provide for web browsers. The Java Activation Framework allows for the querying and handling of multimedia data types.

   As more advanced and richer featured devices become available, JAF will provide the programming framework needed to integrate e-mail application with the functionality of MIME.

3. Other Design Tools

   Eclipse

   Eclipse is Javamail API design software which helps in compiling the java code and forming the final system. All the programming are invented, changed and finalized in this tool.

   The accession to university account with designer's user name and password, since university provides both imap server and smtp server, designing and testing will all operate on these servers.

   Visio

   Produce most of design diagrams which coherently show the flow of the system and system architecture.

   Latex

   Transfer the dissertation version from word to PDF format, which will help to display the thesis clearly.

5.3 Class Description

5.3.1 Console Screen Class

For usability consideration, system provides an operating interface to allow the users to perform.

Descriptions:

It is to create a textual user interface, which has six options for user to select from. User could perform the different tasks. If user wishes to leave the system, simply it just needs to enter character X to exit.

There are six options on the console screen which are:

1. Receive the messages
2. Send the messages
3. Search the messages
4. List the searched result
5. Search again in the search folder
6. Rename and Delete folders

User can input the right number to call the relevant methods to carry out the operation.

Algorithm

All the methods are called from the related classes and its major function is to create the class of display interface of textual email client. After each performance is done, user can go back to the main menu to decide whether to operate on another function.

5.3.2 Retrieve Message Class

It is one of the basic functions in the system, to receive the emails it needs to build the retrieve message class to get the messages from the message incoming server.

Descriptions:

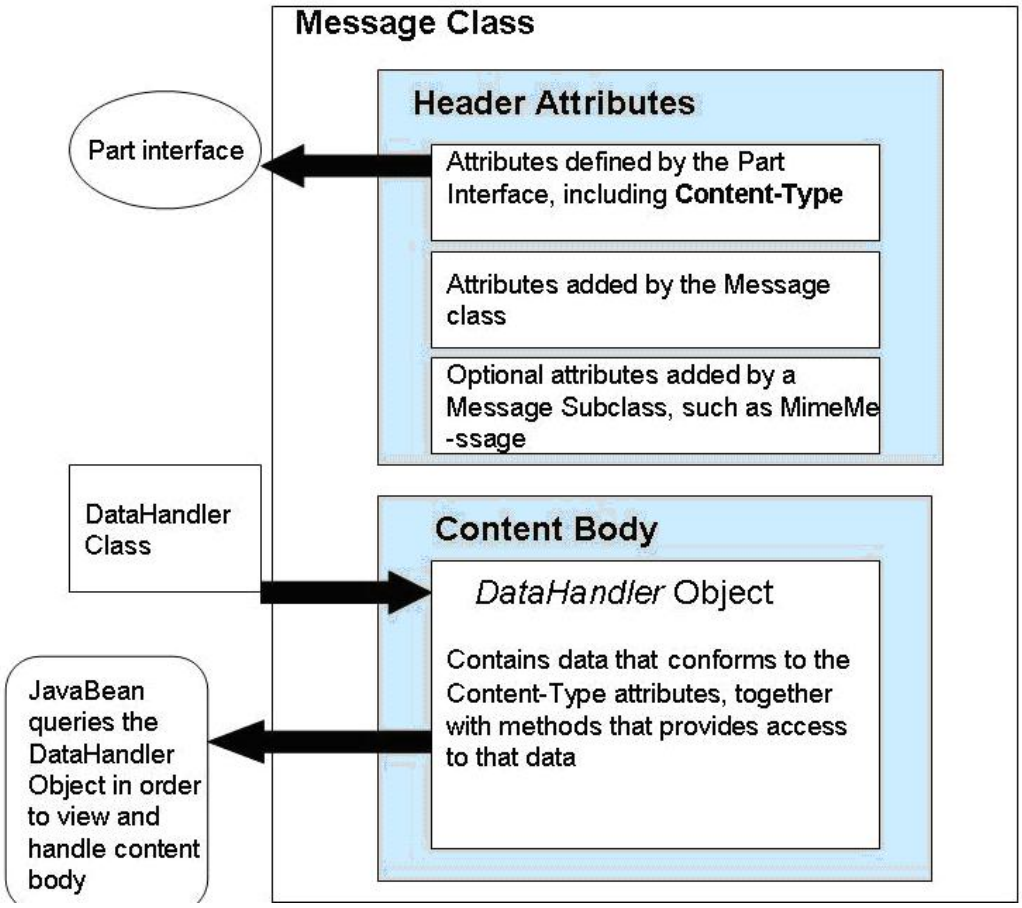

Figure 5.1: Message

Firstly, this is to set the data which will be used across the whole class, which includes all the email header attributes, such as, "from", "date", "subject", "cc", "to" and "flag". Still, it sets the receive server name, user name, password, protocol and default the mail box.

Create the connection to the server which requires the server name, user name and password. It builds up the message properties, session and fetch method to get the messages from the folder.

In the diagram, it shows clearly that the message itself has two parts to design embedding the methods which are specified in Javamail API, the two parts which are the header attributes and content body. So the methods are composed in order to display all the message components.see figure 5.1

Then, the method of asking user to input the address of server, user name and password is produced to establish the connection.

Algorithm

The retrieve message class sets up the connection to imap server by calling the connectImap method and get all the messages in folder. All the message components are created in the MessagePart and MessageBody methods, so user can view them in the console screen.

5.3.3 Send Message Class

Another message performing is to create the send message class to post messages to a group of destination addresses.

Description:

This class is for messages transportation, which has method to connect to outgoing message server smtp and method to construct a message in the console screen.

Then we need to set the messages attributes which are "from", "cc", "to", "subject", "date" and "body", and other smtp data, including smtp property , authorization, and smtp port. For the default protocol port, I set the port is 25.

To construct the message and sets attributes like recipient addresses, inserting content, subject, etc. At the end, it sends the message by invoking the Transport method.

It calls for the user to input all the information in the console screen for an outgoing email. User needs to enter the smtp host name, user name, password, port number, sending mailbox, destination mailbox, cc group, subject and the content of message.

Algorithm:

Send Message class acquires the message properties and session, and then it gathers the user inputting data of message, finally it processes the message to smtp server to accomplish the task.

5.3.4 Search Message Class

This is an important part of email client design; it is to handle the messages search functions and create the virtual folder to save the search results. All the messages which meet the requirements are returned and displayed to user.

Description:

Firstly, it is to set the default mailbox which is the search area for the following search methods. I use the default mail box which is INBOX because it is the term which is specified by Javamail API. Similarly as structure in the retrieving the messages class, we need to set the message properties and session, create the connection methods to the server, and then produce a search folder which will be used to keep the search result. Then next step is to set the message header attributes which contains "from", "subject" and "flag",etc and create the message contents.

Then build up the search string term and each field of searching criteria. The major methods of this class are to create search method.First step, separate a search string like name=***, from=*** to a single search string and call related function for the second splitting in the second step.

The key words will be compared when meets the logical operator AND,OR,NOT.

Afterwards, sort out the different search fields such as from, subject, flag and etc into the search term.

In the end, it calls for the user to input all the information in the console screen for connecting the imap server. User needs to enter the imap host name, user name, password and whether he would like to carry on the search function, once he has submitted, system will ask for the search term which includes from, subject, flag, and string and perform the searching, once the searching is done, the matched messages will be presented to user.

Algorithm:

User takes the search action in regular expression with different logical operators, for example, "From=A [AND B], Subject=course work", thus, search this searching procedure will be performed in the imap server, once messages matching the specified term and will be returned. In this example, search will carry to match the first element is A, once the meets the operator, AND, OR, NOT, system will build up another term B and compare them in the server, the result will be returned after comparison.

```
66              Message[] message=folder.getMessages();
67              if(searchTerm != null) {
68                  message = folder.search(searchTerm);
69                  Date date = new Date();
70                  String folderName = "Search";
71                  Folder searchFolder = folder.getFolder(folderName);
72                  if(!searchFolder.exists())
73                      searchFolder.create(Folder.HOLDS_MESSAGES);
74                  searchFolder.appendMessages(message);
75              }
```

Figure 5.2: Virtual Folder

Virtual folder will be created in this class as well, after checking the search folder existence, system then will import the search message into the folder.see figure 5.2).

5.3.5 ViewSearchFolder Class

The considering functions should be set up for users,[22] to help the users to recognise better than recall, the class is mainly designed to allow the users to see the searched result in the sole search folder, and decide whether to take further search action or not.

Description:

User can view the messages in the search folder which holds all the search results for further use, system will ask the user to check the existence of folder first, once it match the name, system will display the messages within the folder or else error messages will be given.

Algorithm

Class has the message components and calls the method to display in the console screen.

5.3.6 Searched in Search Folder Class

Search performance is limited within the imap server and power of Javamail API, the results returned from the first search maybe contain the false messages or, search is still too wide to view the particular message. To relocate the messages, user would like to search again to get targeted.

Description:

Similarly, class is inherited same search terms which are discussed in the search class, with viewing whether search folder exists from the console screen, user can perform anther query.

Algorithm:

Users are required to define new search expressions to find out the wanted messages; new virtual folders will be set up by the time order which contains corresponding messages.

5.3.7 Rename and Delete Folder Class

As been studies, users make extensive use of folders: Venolia [23] found that average user had 104 folders in the email client, and revealed by Ducheneaut [18] found an average of 90. Whittaker and Sidner [24] found that 35users' folders contained only one or two messages. hese studies identify the necessary management methods of folders should be complies to clarify the email client. I produce this class aiming assist users to organise their folders.

Description:

This class is designed to list all the sub folders and folders within them; user can rename or delete them from the system.

Firstly, crating method is to open INBOX folder as defaulted root folder of the system, and then system displays all its subfolder in order.

As root folder specification, it could not be deleted or renamed. However, user is able to enter these subfolders' number which related to the name of each individual sub folder. Afterwards, system displays sub folder with its folder inside, and then user could rename or delete it by entering correct option number. Once folder is renamed, new folder name will be displayed, on another hand, after the deletion, system will go back to previous folder to allow user to do other performance.

Algorithm:

Methods are built to connect to the imap server and search in the root folder to display all the sub folders. To rename the folder, firstly, user inputs a new folder name on console screen and then system will copy the messages from current folder to new folder, where after, system deletes current folder. Delete action will be operated in server directly.

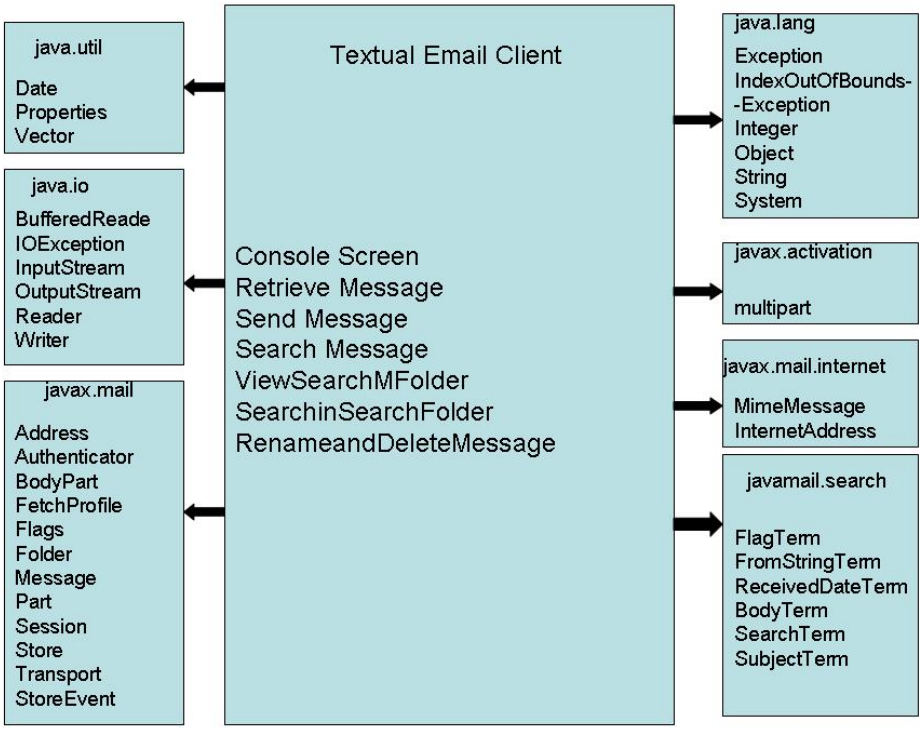

Figure 5.3: Classes and Libraries

5.4 Diagram of all the classes and libraries

5.5 Further Implementation from Design

Detail design is a re-thinking and modification process, new ideas are bought in and some requirements are discarded in the actual design, the real system will be developed with class adjustment, I will give details what the new functions are and why they are developed.

5.5.1 Interface Implementation

At the very beginning, I composed the entire system in a sole programme which sets two arguments -R and -S in the main method to perform receive, send and search functions.

Soon I find it's hard to adjust the entire system and inefficient to implement the programme. So I separate classes to achieve the different missions, which are specified in the chapter four. Though all the messages can be displayed in the console screen and user manipulates the system in command lines, it requires an interface to avoid reluctant information and repeating some routine inputting staff, what is more, user will be clear with instructions in the interface.

All the steps of different behaviors are listed in the console screen, once user wants to leave the system, they just enter the X character, otherwise, selections of other characters or wrong numbers, system will display an error message to remind user and ask him to try again.

5.5.2 Monitor Folder

To fetch messages in the folder and display to user, is prior version in first design development. Because the default folder is INBOX, I assume the folder should contain a certain amount of messages, but there are weaknesses in the consideration which are empty folder, invalid folder, no default folder and messages overloaded.

For the first one, if there is no message in the folder, system should give relevant information to user as friendly reminder. The other two, if system can not find the default folder which is specified, error messages will be given. The later one, all the mailboxes have the maximum memory to store messages, once messages exceed the folder limitation; the system is responsible to tell user the over loaded messages see figure 5.4.

5.5.3 Error Message

Sometimes user may come across the situation such as unexpected application error or performing illegal operation. User should be informed with error message which gives polite, concise, consistence and constructive warnings of how to correct mistakes.

Based on the user's skill and experience, kind error message is displayed in console screen, for example, if user inputs characters instead of numbers in the selection, system would remind him to input right number and try again.

But since I only implemented system in several classes with error messages, further work will be required to complete the system.

```
73              Folder folder =store.getDefaultFolder();
74              if(folder==null){
75                  System.out.println("No default folder");
76                  System.exit(1);
77              }
78              //if the defaul folder can't be found
79              folder=folder.getFolder(mbox);
80              if(folder==null){
81                  System.out.println("Invaild Folder");
82                  System.exit(1);
83              }
84
85              //To calculate the messages within the folder
86              //once on message found, system displays empty folder
87              int totalMessage=folder.getDeletedMessageCount();
88              if(totalMessage==0){
89                  System.out.println("Empty Folder");
90                  folder.close(false);
91                  store.close();
92                  System.exit(1);
```

Figure 5.4: Folder Monitor

5.5.4 Folder implementation

Specific requirements are built after literature review, but new problems emerge in the actual design.In the search class, system creates only one search folder to save the result to avoid the folder overloading problem[24], at each time user runs the search function, search folder will be renewed and new messages will be sorted into the folder. But, since each time folder has been modified, user will be confused which messages according to search result every time, further more, it occupies much space of email client and it should be removed from the system.

There are solutions which are aimed to solve these troubles:

1. Sub Virtual Folder

   Since all the searched messages are grouped together in a single search folder, or the search at the first time may still has too many possible emails, in this condition, user may feel like running the search criteria again in the search folder, so sub folder will be created to matched result and rooted in a tree hierarchy.This hierarchical virtual folders improves the searching function more effectively.

2. Folder Rename and Delete

Folder rename will result in comprehensive query at each time of sub folder creation; user can rename it based on their understanding which is initially produced by system in time order. In addition, user could delete the folders which they regard there is no value and reluctant except the default folder INBOX.

5.6 Summary

Implementation is to set up the final system and show the system availability, how to implement and finalize the system is discussed in this chapter. In the language selection, I choose the java language which as stated in Java language white paper by Sun Microsystems: "Java is a simple, object-oriented, distributed, interpreted, robust, secure, architecture neutral, portable, multithreaded, and dynamic." The fact confirms that it is the right decision, and then I go to each class description which is inherited from design approaches. Nevertheless, system has the variations from the original design assumptions; I state them with explanation why these changes are made.

Chapter 6

Testing

6.1 Introduction

After the detailed design and system implementation, testing should be followed to exam whole system's consistency, usability and integration, etc. It is to identify the potential problems and correct the errors within it. The testing methods and test fields will be carefully selected to reflect the each function in system, as been discussed in the non-functional requirement; system should be reliable and well behaved in the usability requirements. Finally, in the comparison with other email clients, it will demonstrate my email client's feature and weakness to be improved in future use.

6.2 Test Plan

Test plan is constructed before going to the actual testing, it will form the areas the testing and which prescripts I want to perform and how the problems will be identified. For the plan, I will take the following into account:

6.2.1 Preparation

This will ensure that all the reasonable aspects running at a test have been considered.

6.2.2 Communication

It will help when I invite other users to perform on the system, but the detail operation instructions should not been revealed or else it will affect feedback correctness.

6.2.3 Effectiveness

[25] This is supposed to provide a mechanism for outlining test needs, limitations and justification for purposes of setting expectations, acquiring resources, investigating unexpected results, assuring normalcy and effectiveness. Based on my system, different testing methods will be adopted to throughout the entire components which include system, functions, interface and comparison with other email clients.

6.3 System Testing

System testing is concerned with testing the entire system, where I will use the Release testing which is concerned that system meets its requirements, and to ensure that system is dependable. Release testing is usually "black box" testing where the system is to tested on working performance properly or not.

6.3.1 White Box Testing

White box testing is to test the system internal structure which is also known as glass box testing or structural testing. Without acknowledgement of programme arrangement, white box testing is applied to test the code of the programme to examine its output whether is appropriate. Testing will be performed only if knowing what the programme is supposed to do. Tester will see if the program diverges from its intended goal. White box testing does not account for errors caused by omission, and all visible code must also be readable [25] Since the system will be designed, implemented and tested all by author, the code testing has initially been done in the constructing the system process so it is not necessary to carry out the white box testing to examine the programme code again.

6.3.2 Black Box Testing

Black box testing technique is used to select test cases while as white box testing is mainly applied to determine whether two objects resulting from the program execution of a test case are observationally equivalent. The reason why I choose black box testing is system is produced and tested both by myself, it's not necessary to "peek" to internal structure of the system, besides, the specification is produced once the requirements are established, so, it's more appropriate to apply the black box testing to see whether system has conformed the published requirements.

[28]My textual email client, the input is from the keyboard and output is computer's display. To evaluate the entire system, every class will be checked by using black box strategy. The function in these classes will be tested as a unit examine to see whether it meets the desired functionality.

Black box testing behaviour is determined by studying the input and related outputs. The following section is adopted this method to test the system.

Input and Output

Accordingly test area will comprise all the functions in the system, the input values are specified as requirements and output will be system displays.

Some guidelines of designing the input and output are:

1. Design inputs that force the system to generate all error messages;
2. Design inputs that cause input buffer to flow;
3. Repeat the same input or a series of input numerous times;
4. Force invalid outputs to be generated;
5. Force computation results to be too large or too small;

Consequentially I design the input values based on these selective guidelines and the detailed inputs and system outputs will be put in Appendix Two.

6.3.3 Usability Testing

Usability involves a couple of system requirements which are effectiveness, learn ability and flexibility. A system's usability relies greatly on its functionality. Usability testing consists of a variety of methods for assessing the usability of products and product designs.

Since usability testing is concern about interface testing and user manipulation, I invite a small group of students to operate on my system. They are all from university as "heavy users" with email using experience and relevant knowledge. Before they get contact with system, I simply explain the system is textual email client, and its main approaches are virtual folder and multiple searching functions.

The first problem emerges when they practice the system is password visible on console screen, although it is for individual use, user still feels like it to be protected in the system. They suggest that they want their personal information be stored in the system to save time when they log in next time. During the process of following the console screen instruction, they all are clear what it is about and can carry out the tasks as been introduced.

In the receiving and sending messages, positively I observe that they all can perform properly, but they are less happy about not many relatively error messages given from the system when mistakes are made and alternatively, the system is expected to have "help" option to assist users using it. They regard the messages which displayed from the system are complete and easy to read from.

Importantly, for the feature of searching function provides, they all can use appropriate searching expression to locate messages which meet the criteria, the returned messages displayed from virtual folder they think cover the area they want, I ask what they expect for further extension in searching, they consider the precision matters more than the quantity of searching result.

They also can execute the further searching within the search folder and view the result as well. They think the system achieves the initial design goals but further implementation can be applied to complete the system.

6.4 Compare with other email clients

After the implementation of the system, to demonstrate the system feature and identify the weakness, I compare my Textual Email Client to some other email clients to show the differences clearly:

Email Client	TEM	Outlook Express	Mozilla
Portability	+		+
Perform Search Express	+		
Flexibility of using logical operator	+++		++
Folder Management	+++	+	+
Interface Using	+	++	++

6.4.1 Portability

[29]Portability which means the ability of using the email client in different platforms, users can operate on the email client despite the environment changes.

My system is produced and implemented in java language, it has advantages that it is platform independent and allows user to receive and send emails by connecting to server. Users can use this system in any platform still.

6.4.2 Perform Search Express

With efforts of creating search function in the system, now it can use regular expression in the system to do the single or multiple searches. User can set the search area and get the correspond messages. It increases the efficiency on message operation.

6.4.3 Flexibility of using logical operator

This is to use to different combinations of logical operators to search on the email header and body part.

My system embeds the logical operators which are "AND", "NOT", "OR" with brackets to form a variety of search query expressions, the search result will be well located according to using these operators.

6.4.4 Folder Management

By creating the virtual folder, messages can be well sorted out to different searching sub folders which will help users to organise their messages and further management of folders also can be performed such as deletion and rename.

6.4.5 Interface Using

My TEM is text-based interface, although it does not interact with users in high level compared to graphics email client, it still has advantages in plain screen selections in which user will read and write messages easily without too many buttons confusion, more over, it is fast to perform since it will not be bothered with HTML messages displaying and decoding.

6.5 Summary of Testing Result

To select right testing methods is the first step in testing practice,more importantly, what are discovered from the testing and what their meanings are should be sum up. The testing major purpose is to discover system potential problems and show the system reliability of use. From all the testing areas of being done, I would summarize the result as:

6.5.1 Achievements:

1. System would be operated under current condition and users have no problem in manipulating it;
2. System can perform all the functional design goals which are receiving and sending messages;
3. System can reach main design aims which are virtual folder and multiple searching performance;
4. System is stable and consistent to use and no corruptions are revealed;

6.5.2 Weaknesses:

1. Security issue: I have not solved the problem of making the password invisible in system so system is only provided for individual use whereas user enters password each time because system has no memory to keep it;
2. Not enough informative feedback:[30] user should see the consequence of their actions, but my system does not have many error messages may lead user confused and disoriented.;

3. Searching performances: the searching algorithms are used which javamail API supplies, but they are not perfect since the result contains some false messages

Both achievements and weaknesses are valuable aspects of doing this system design; further ideas can be generalized based on these recognitions, the redesign or further improvements can be built based on these identifications.

Chapter 7

Conclusion

Back to introduction of thesis, it has introduced background information about what it is of email client, and then it has discussed one of the highly demanding improvements is to help users to organise and find the wanted messages. This forces to explore efficient solution which is my project design goal.

System will be determined by using multiple search expressions and virtual folders to achieve the design aspiration. The approaches will be carefully studied and evaluated. The following chapters are formed enclosing the project design process.

Literature review gives a large scale of system designing background learning. Talking about the email client advantages as well as problems is to make users have more clear pictures of the necessity of studying and improving the email client. Besides, some basic information also is provided, after these theoretical issues, I highlight my design scope which is to build up virtual folder and store the queries in different combination searches. The following of evaluating other email clients I have stated current email clients whether they have coped with problem, from the estimation, it points out the weakness of current problem solution. I proclaim the design language I would use and what kind of search I want to perform.

But when I finish the system design, back to the literature review, there are some reflections upon it. The literature review sets up initial design frame of system, the design aim is still straight-forward during the whole system design process. I use the Javamail API language as specified in literature review to build the text-based email client, the design goal remains focusing on searching queries and creating virtual folder.

Both of them have been achieved in the actual system, but there are some adjustments should be made when applying in the actual design to be alignment and make system complete and stable.

The main improvement is to create the sub virtual folder in order to precise the searching result from search folder. In the literature review, simply I just stress on the virtual folder constructing, but actually, we all are experienced in getting too much information from once searching result. Still, it is inefficient to create many search folders in the email client which will cause folder over flows, so I only set up one search folder and refresh it each time, it leads to another problem, which is search folder contains too many messages.

So, sub folder creation is carried to improve the system performance, it is a sub virtual folder rooted in search folder which minimize the search target. The design details are introduced in the Chapter five, detailed design and implementation.

In next chapter,The scope of the system will be identified, system will be well managed by the precise functions of what it will achieve. The requirements acting like the measure standard of what can be done, studying from different resources, I tackled the functional requirements of what the system is expecting to perform, user and the interface requirement are part of it which governs the project from the view of users. With non-functional requirements, it sets constrains on system operating environment and its development. These considerations will be concern about the system robustness, extensibility, security, etc. In reality, a system can not be "perfect" and it must focus on some problem clarifications instead of trying to reach every aspect of the matter, it is tedious and unrealistic. As a result, the requirements specification compared with original requirements with actual system design to structure system design functions.

System design is referred to the functions discharged from requirements evaluation; different models are used:

1 The first model,client-server model shows how the system interacts with internet and connects to messages servers;

2 The second model, which is single system module, it shows the potentially each entity of system have the methods and how these entities linked to each other;

3 The following models are what I assume how the each task is carried in the system;

4 Finally, from search actives, whole system picture is generalized on the approach of performing the entire system.

Showing the design strategies is the key part of the design. The emphasises are upon the multiple search and folder criteria. I investigate the search terms covering almost all the components of message with careful expurgation. With using of diverse combinations of logical operators, users can express the query in regular phrase. Virtual folder will be set up originated from the default root folder, containing search result, the manipulation messages within in will not affect the existence of root folder.

Concrete system design is followed, using selectively design language and platform, system classes are programmed, with description and algorithm, users will find it is quite straight-forward to understand. But actual design is a recycle progress which might variate from the design suppositions. At the end of the implementation, I listed several supplements in order to make system better to use.

In the testing section, sole testing method can not consider all the aspects of system. System testing, usability testing and comparison with other email clients are used. The testing result reflects that, my email client has met all the design goals although it has weakness in security issue which has not solved the password visible problem.

7.1 Furture work with Possible Solutions

To develop a good software needs considering more that one or two aspects, since my design still has develop space in future work, better and more powerful work can be produced.

Since I apply the search engine in javamail API which is done by matching the incoming messages against the query which is associated to folder. The search result will be added into a folder, performance is good to search in hundreds of messages without noticeable delay. A list of messages contained in a folder according to particular query, but the fact is the folder might contain false emails, the demanding further implementation would be a message reaches certain cut-off score before sorting into the folder, or, search only be applied in new messages to the top one or top few matches.

So a better classification mechanism should be brought in at the direction of sorting messages into categories relating to the topics, at this stage, reliable and more accurate automation detection should be designed. Integrated with other dominant search engine or languages would get better search performance.

The textual email client is not complete; it still has other features can be added in to consummate the system. The components can be viewed both from the Chapter three of requirements and Chapter six testing. Major demanding improving are attachments association, reply to sender, forward message, processing HTML, etc.They all can be solved in current system.

Throughout these future work assumptions, email client will be further implemented and developed, a clear field to extend the work is to pay attention to the usability enhancement, user will appreciate the friendlier and communicative interface and considering help conditions as system provides appropriate feedback in a reasonable time [22]. With more interactive interface, user would be clearer how the system is to be used. It will be beneficial while user are more likely to use the system which has fast and plain screen along with sufficient and robust search functionality.

System reliability, consistency, efficiency and flexibility will all be carefully taken into account. Some of the functions may be more relative to develop while some will be added as another system's attributes in the search fields. Under the design recycle route, with summarized advantages and inefficiencies, new version of system would be released with better capabilities.

Email client is basic use software now widely used by different groups of people and for different purposes. Since communication is much changed by emerging of email, how to make email client assist people better, it is still an interesting and questioning area. My approaches in this field are a direction of studying message organising. I believe that, this work will be investigated widely, powerful and flexible search-based email client has high level of competition in future market.

Bibliography

[1] http://www.domainsarefree.com/glossary/Mail_client.html, *Mail Client*,[Access Date: May, 2005];

[2] Ethan Phelps-Goodman and Matthew Chalmers, University of Glasgow, *QueryMail: Using Search to Organize Email*, September, 2004

[3] Kidd, A. The marks are on the knowledge worker. *In Proceedings of CHT94 Human Factors in Computing System*, 186-191. ACM Press, New York, 1994.

[4] Lansdale, M, *The psychology of personal information management.* Applied Ergonomics, 19, 55-66, 1988.

[5] Luke McDowell, Oren Etzioni, Alon Halevy, and Henry Levy, *Semantic Email*, University of Washington, May, 2004.

[6] Lorrie Faith Cranor, Brain A. Lamacchia, *Spam!*, August 1998, Communications of the ACM.

[7] Venolia G D, Dabbish L, Gupta A, Cadiz JJ (2001) *Supporting email workflow*, Page 4-5, September, 2001

[8] Bernard Kerr, Eric M.Wilcox, *Designing Remail: Reinventing the Email Client Through Innovation and Integration*, April 24-29, 2004, Vienna, Austria, ACM 1-58113-703-6/04/2004.

[9] Hal Berghel, *Email-The Good, The Bad, and The Ugly*, Communications of the ACM.

[10] M.Crispin, *RFC 1733*, University of Washington, December, 1994.

[11] Victoria Bellotti, Nicolas Ducheneaut, Mark Howard, Ian Smith, *Taking email to task: the design and evaluation of a task management centered email tool*, April 2003.

[12] Victoria Bellotti, Nicolas Ducheneaut, Mark Howard, Ian Smith, Christine Neuwirth, *Innovation in Extremis: Evolving an Application for The Critical Work of Email and Information Management*, June, 2002.

[13] http://www.sun.com, *Javamail API documentation.*[Access Date: December, 2004]

[14] http://gmail.google.com/gmail/help/about_whatsnew.html,*Gmail Introduction*,[Access Date: December, 2004]

[15] http://www.mozilla.org/support,*Thunderbird Guide*, [Access Date: December, 2004]

[16] Streve Whittaker, Candace Sidner, *Email overload: exploring personal information management of email* , April, 1996

[17] Ian Sommerville, *Software Engineering*, 6th Edition, Chapter 5 Software Requirements, 101

[18] Victoria Bellotti, Nicolas Ducheneaut, Mark Haward, Christine Neuwirth, Ian Smith, Trevor Smith, *FLANNEL:Adding computation to electronic mail during transmission*, October, 2002

[19] Ian Sommerville, *Software Engineering*, 6th Edition, Chapter 10 Architecture design, 222, 229

[20] http://www.weblicon.net/html/arch_messg_architecture.html*Messaging Architecture*,Access Date, May, 2005

[21] http://tutorials.freeskills.com/read/id/60, *PROFESSIONAL WAP PART 3 - INTRODUCTION TO THE JAVAMAIL API*, [Access Date, May, 2005]

[22] http://www.useit.com/papers/heuristic/heuristic_list.html, *Ten Usability Heuristics*, [Access Date, April, 2005]

[23] Venolia, G.D, Dabbish, L.,Cadiz, J.and Gupta, *A. Supporting Personal Information Management of Email Workflow. Microsoft Research Tech Report*, ftp://ftp.research.microsoft.com/pub/tr/tr-2001-88.pdf, 2001

[24] Whittaker, S. and Sinder, C.*Email Overload: Exploring Personal Information Management of Email*. Proceedings of Conference on Computer Human Interaction, CHI'96, 1996

[25] http://www.isixsigma.com/library/content/c040920a.asp, *Importance of Test Plans/Test Protocol (with a Template)*, [Access Date, May, 2005]

[26] http://www.webopedia.com/TERM/W/White_Box_Testing.html, *White Box Testing*, [Access Date May, 2005]

[27] Ian Sommerville, *Software Engineering*, 6th Edition, Chapter 20 System Testing, 440

[28] Ducheneaut, N. and Bellotti, V. *Email as Habitat*. Interactions, 85, 2001

[29] Chan, Yuen Pang, University of Bath, *Email client prototype 'ECP' with advanced query features*,2003

[30] http://studiolab.io.tudelft.nl/dekoven/stories/storyReader$9, *Schneiderman's Golden Rules*, [Access Date May, 2005]