December 2002

DEVELOPING AND IMPLEMENTING THE MAIL CHECKER FOR ELECTRONIC MAIL CONTROL

Seong-No Yoon
University of Nebraska - Lincoln

JongHeon Kim
University of Nebraska - Lincoln

DEVELOPING AND IMPLEMENTING THE MAIL CHECKER FOR ELECTRONIC MAIL CONTROL

Seong No Yoon and JongHeon Kim
University of Nebraska – Lincoln
syoon81@unlserve.unl.edu Jkim9722@unlserve.unl.edu

Abstract

Mail Checker is an email-checking tool written by JAVA program (SDK 1.3.1 and SWING). Mail Checker has the following functions: retrieving the header information (i.e., "From," "Subject," "Date") of e-mail messages from POP3 servers, sending e-mails using SMTP server, and maintaining the user's e-mail accounts. In virtue of portability, one of JAVA properties, Mail Checker can run on Unix-based platforms as well as windows-based platforms. In addition, the tool enables a user to select an e-mail account at his/her disposal if he/she has more than two e-mail accounts. To efficiently integrate all of user's e-mail accounts, this tool provides an integrated repository to efficiently maintain all of the user's e-mail accounts. Through this repository Mail Checker can navigate POP3 servers registered by a user.

Introduction

Electronic Mail (E-Mail) has become a primary mode of communication in the age of connected computer networks. E-mail also makes exchange of information possible among unknown people as well as well-acquainted people. For example, one can get some help from experts or professors through the Internet, by sending questions through e-mail. Constant et al. (1996) found that technical advice through e-mail from distant employees on the electronic weak ties was useful.

Nowadays many information providers supply free e-mail accounts for their customers or just applicants. A flood of e-mail accounts can cause people to have a big problem of maintaining their accounts. To check messages from e-mail accounts a person has, one must log in each e-mail account, while struggling to remember their login identifiers and passwords. Many e-mail checking software such as Microsoft Outlook Express can set people free from memorizing their e-mail accounts they have. Such software, however, does not run on Unix-based platforms other than Windows-based platforms without change of source codes of the software. Coding in JAVA gives great portability to the software, which in turn provides numerous applications of the software. This Mail Checker not only runs on Windows-based and Unix-based platforms but can also be implemented in mobile appliances such as PDA and PCS.

Many people use multiple e-mail accounts to personalize their mail. One account may be used for receiving messages related to advertisement of companies, while another is for friends or families. Most e-mail checking programs require setting an e-mail account as default for sending and receiving e-mail messages. Default e-mail account registered in the software takes care of sending e-mail messages. For example, if a person has set an e-mail account provided by Yahoo as one's default, then all e-mails sent are shown with the sender's address of Yahoo account. On occasion, it is necessary to change our e-mail accounts when we send an e-mail using the e-mail checking programs.

Mail Checker is an email-checking tool written by JAVA program (SDK 1.3.1 and SWING) (Booch et al. 1999, Deitel and Deitel 1999, Douglas 2001, Savitch 1999). Mail Checker has the following functions: retrieving the header information (i.e., "From," "Subject," "Date") of e-mail messages from POP3 servers; sending e-mails using SMTP server; and maintaining user's e-mail accounts. In virtue of portability, one of JAVA properties, Mail Checker can run on Unix-based platforms as well as windows-based platforms. In addition, the tool enables a user to select an e-mail account to be used for sending e-mail at one's disposal if he/she has more than two e-mail accounts. To efficiently integrate all of a user's e-mail accounts, this tool has an integrated

repository of all of the user's e-mail accounts to efficiently maintain. Through this repository Mail Checker can navigate POP3 servers registered by a user.

Tis paper is organized as follows: Section 2 presents protocols of POP3 and SMTP; details of the implementation are described in Section 3; Section 4 presents discussion; finally Section 5 provides conclusion and suggested future research.

Background

RFC 1725 is a document by the Network Working Group, entitled "Post Office Protocol - version 3" (POP3) (Myers and Rose 1994). This document specifies Internet standards track protocol for the Internet community and includes in detail how POP3 functions. To communicate with POP3 servers, a client (program) needs to use its port number of 110. That is, a client requests mail messages on the port number and receives the messages through the port number. Protocol for communication between a client and POP3 server is shown in Figure 1. POP3 server listens to port number 110 to catch a client connection request. If a client requests connection, then the server responds with the message of "+OK…" After the client sends "USER *userid,*" where *userid* is the client's login id, the server requests the client's password. After verification of the client, the server sends out all messages. All bold typed characters are predefined commands in POP3 protocol.

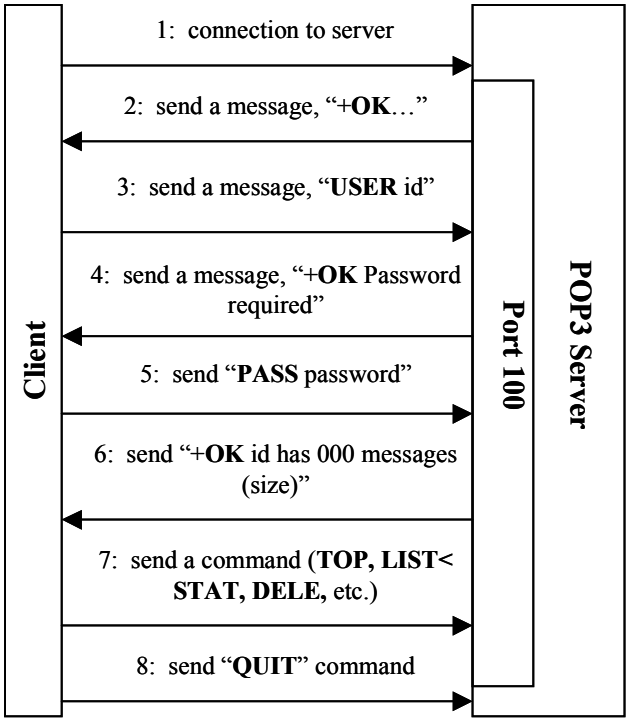

Figure 1. Protocol between a Client and POP3 Server

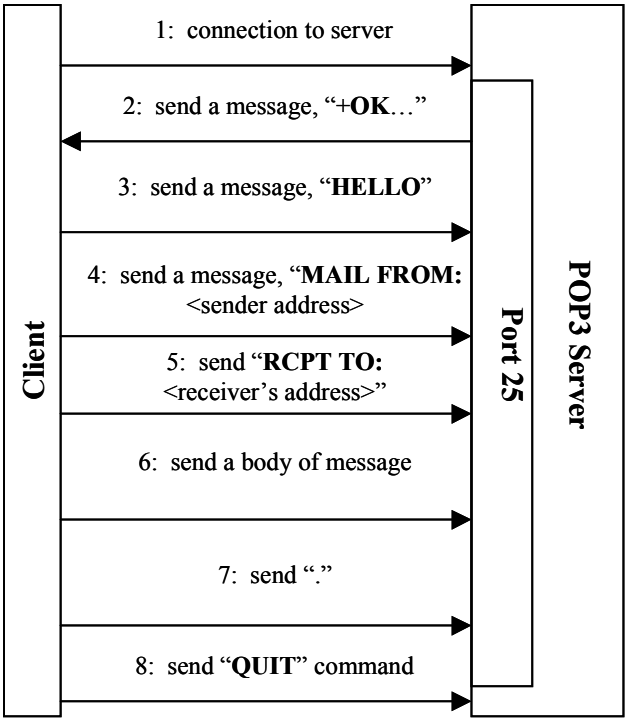

Figure 2. SMTP Protocol

Simple Mail Transfer Protocol (SMTP) is used to send a message, using a port number of 25 (Douglas 2001). The protocol for communication between a client and the SMTP server is shown in Figure 2. After a client's connection with a server, the client sequentially sends "HELO" command, sender's e-mail address, receiver's e-mail address, and message. It is noted that the client should send "." as a sign to end message.

Design of Mail Checker

Mail Checker provides three main functions: checking e-mail messages from POP3 servers, sending e-mail messages, and maintaining a user's e-mail accounts. The system structure of Mail Checker is shown in Figure 3. Mail Checker consists of three

main classes: RMailPanel, SMailPanel, and SystemPanel. RmailPanel performs checking e-mail messages received and fetching the mail headers (i.e., "From", "Subject", "Date") from POP3 server. SMailPanel sends e-mail messages, following SMTP protocol. SystemPanel allows a user to register his/her e-mail accounts and manage them. All accounts data are stored in the CONFIG file. Mail Checker is initiated from the starting point, MailChecker class. Using the accounts information of the CONFIG file, Mail Checker can validate authorized users. Given a user's account and the choice among three functions, MailChecker class initiates a class correspondence with the function choice of the user, which returns with a status of job (i.e., success or fail).

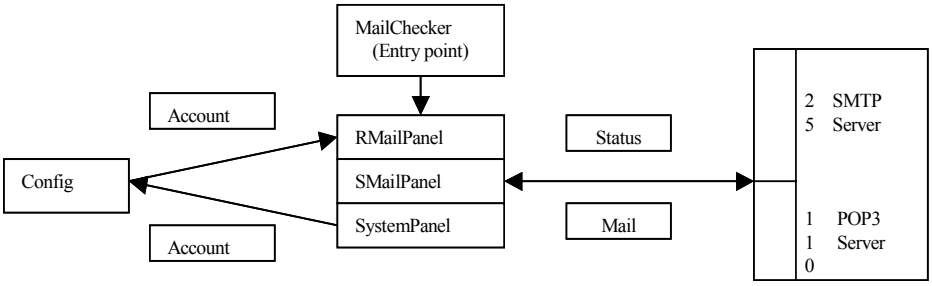

Figure 3. System Structure of Mail Checker

When accessing POP3 servers, RmailPanel, SmailPanel, or SystemPanel class calls for the status class and Mail class. Status class keeps track of the results of POP3 commands performed during a session. Using these results, Mail Checker can figure out whether POP3 commands run correctly or not. Mail class stores mail headers ("From," "Subject," "Date") of e-mail messages from requested accounts. The mail header information will be shown as a result of mail receiving by RMailPanel class.

The main methods of each class in Mail Checker are shown Figure4. This class diagram is drawn using Rational Rose software (Booch et al. 1999). A total of nine classes corroborate with each other to operate the Mail Checker tool. RmailPanel, SmailPanel, and SystemPanel class have a dependency relationship with the MainChecker class.

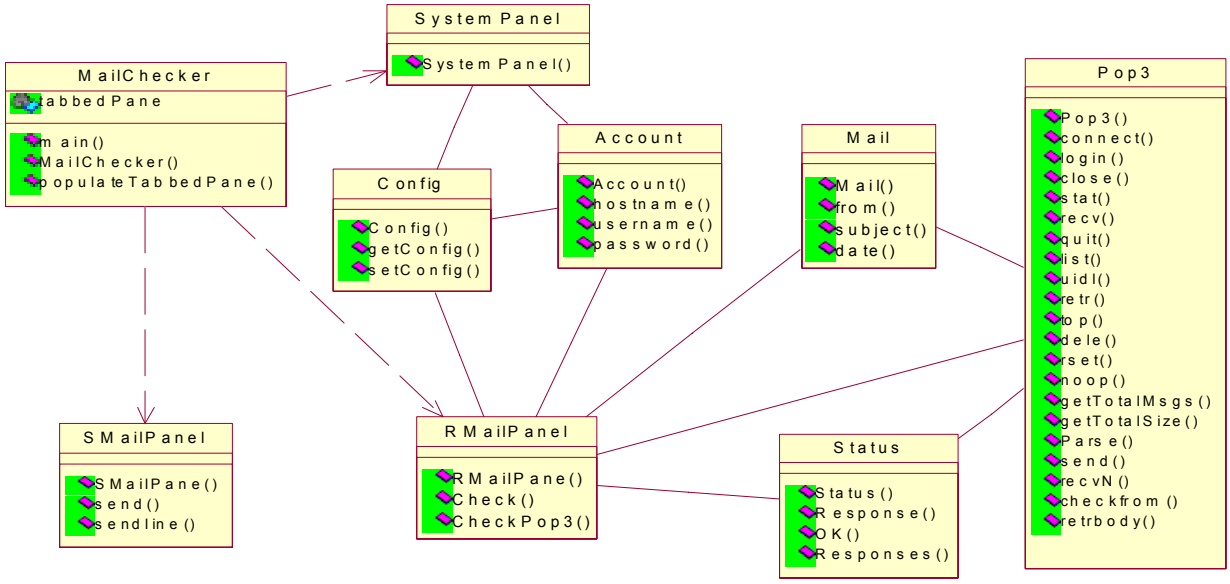

Figure 4. A Class Diagram for Mail Checker

MailChecker Class

This class is the entry (starting) point of the Mail Checker tool and shows initial screen with three menus: Mail, Compose, and System. The screen is based tabbedPane calss, which is one of classes supported by JAVA SWING class. The Mail menu functions to retrieve mail header information from registered e-mail accounts, while the Compose menu enables a mail message to be sent. System menu manages users' POS3 servers, user IDs, and passwords, enabling to insert, update, or delete them.

RMailPanel Class

The RMailPanel class retrieves headers of received e-mail messages in each e-mail account registered in Mail Checker and displays header messages on the screen. Figure 5 shows the results. RMailPanel class perfroms as the follows:

1) Looks up a POP3 server ID and its account information in CONFIG file
2) Connects to POP3 server using the Account information
3) Gets the mail headers (From, Subject, Date) from each POP3 server
4) Displays results on the screen
5) Repeats 1) - 4) until all of registered accounts in CONFIG file are processed

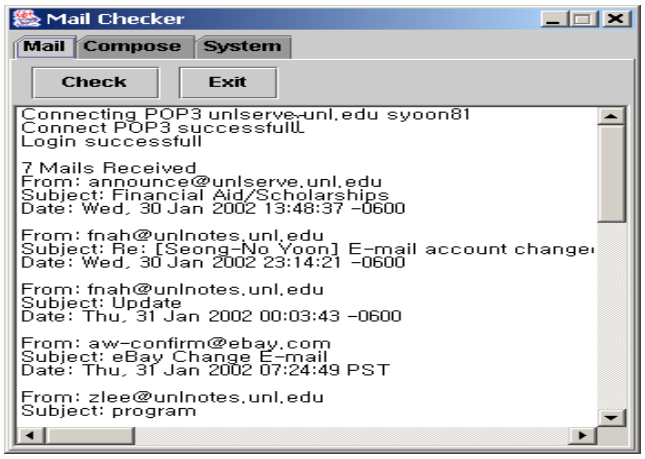

Figure 5. A RmailPanel Class

SMailPanel Class

The SMailPanel class sends an e-mail message from any e-mail account using SMTP server, which listens to port number 25. The protocol for communication with the SMTP server is implemented in this class.

Connects to SMTP server using port 25
1. Sends "HELO",
2. Sends "MAIL FROM: ",
3. Sends "RCPT TO: ",
4. Sends "DATA",
5. Sends e-mail message,
6. Sends "." in this sequence.

SystemPanel Class

SystemPanel Class enables a user to register all of his/her e-mail accounts. Figure 6 shows a screen of Systempanel and message after adding two POP3 servers. Accounts information is written into the CONFIG file through Config class when the user presses an apply button. This class has following functions:

1. Reads the Account information from Config class
2. Displays the current account information on the screen.
3. Adds hostname, username and password
4. Sends the account information to CONFIG file using Config class.

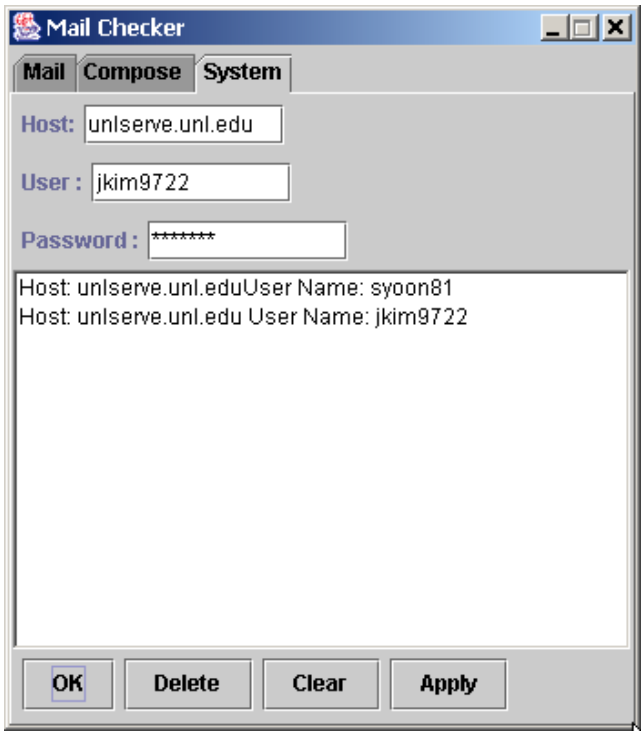

Figure 6. A SystemPanel Class

POP3 Class

POP3 class communicates with the POP3 server storing a user's e-mail messages. The protocol for communication with the POP3 server is implemented in this class. The protocol is based on the RFC 1725 definition for the POP3 mail. The following methods and functions are implemented through POP3 server commands.

Methods:

Constructor:

void	pop3(mailhost, user, password)	

Public Methods

Status	connect(mailhost)	// connect to pop3 host
Status	login(user,password)	// login
Status	stat()	// get message number and size
Status	list()	// list
Status	list(msgnum)	
Status	retr(msgnum)	// retrieve the message body
Status	dele(msgnum)	// delete the message
Status	noop()	
Status	quit()	// disconnect
Status	top(msgnum,numlines)	// get the top lines
Status	uidl(msgnum)	
Int	get_TotalMsgs()	// return number of mail messages on server

Private Methods

void	send(String cmd)	
String	recv()	
Void	recvN(Status status)	

Status Class

Status class deals with returned messages from the POP3 server during retrieval of messages in order to check whether the POP3 server works properly or not.

Public Data

```
boolean_OK = false;                  // True if last command returned +OK
String  _Response;                   // Set to initial response from server
String[] _Responses= new String[0];  // Set to last multi-line response.
```

Method:

```
Responses()   // Return the multi-line output from a command
Response()    // Return the initial status line output from a command
OK()          // Return the completion status (+OK true or -ERR false) from the last command issued
              to the server.
```

Mail Class

Mail class is used to store the headers information of e-mail messages, which consists of "From," "Subject," and "Date," retrieved by RMailPanel class and POP3 class.

Public Data

```
String from;     // True if last command returned +OK
String subject;  // Set to initial response from server
String date;     // Set to last multiline response.
```

Method:

```
Responses()   // Return the multi-line output from a command
Response()    // Return the initial status line output from a command
OK()          // Return the completion status (+OK true or -ERR false) from  the last command issued
              to the server.
```

Account Class

Account class performs conveying the POP3 host information. It has hostname, username, and password. SystemPanel class has Vector array of this class to manage the whole accounts of POP3. RmailPanle class uses this to connect to all POP3 servers, which are registered.

Public data:

```
String _hostname;   // Pop3 host
String _username;   // user id
String _password;   // password
```

Method:

```
Hostname();   // return _hostname for this account class
Username();   // return _username for this account class
Password();   // return _password for this account class
```

Config Class

Config class creates a physical sequential file, CONFIG, into the hard disk running on Mail Checker to save the account information or retrieve account information from "CONFIG" file. RMailPanel, SystemPanel class calls the getConfig() and setConfig() methods. When getConfig() method is called, it reads the "CONFIG" file and saves the accounts information when setConfig() method is called.

Constructor:
Config() // Read POP3 accounts information from "CONFIG" file

Public data:
Vector accounts = new Vector(100) // store POP3 accounts information

Method:
getConfig() // returns the currents accounts data
setConfig() // save new account data

Discussion

Mail Checker has relatively few functions compared to other commercial e-mail checking programs. However, this tool has a special feature, which enables sending out mails using any e-mail account, registered in Mail Checker. Before we designed and implemented the Mail Checker tool, we expected this tool would have better computer memory efficient than commercial e-mail checking software. However, the amount of memory occupied by this tool's execution module is almost the same as that of Microsoft Outlook Express. The amount of memory used by two programs was around 13 megabytes when run on Microsoft Windows 2000. Mail Checker needs to be upgraded for better features. First, it needs to retrieve the message body (content) of e-mail like a commercial e-mail checking program. Mail Checker is equipped to perform a retrieving message body, yet we need to append some special techniques if mail type is MIME or Base64 Code. Second, to improve performance of Mail Checker, it would be better to use threads rather than processes technology to access POP3 Servers. Finally, Mail Checker needs informed user interfaces.

Conclusions and Future Research Needs

ail Checker is implemented using the JAVA programming language and equipped with the following functions: retrieving the mail headers from POP3 servers; sending e-mail through SMTP servers; and maintaining user e-mail accounts. SWING was used to implement the user interfaces, while SDK version 1.3.1-02 is used to develop Mail Checker. Mail Checker can access multiple POP3 servers and show the mail status for each POP3 account. This allows the user not to log on to check the mail in every account. It can run on Unix-based platforms as well as windows-based platforms by the virtue of JAVA language characteristics. Mail Checker is an e-mail checking tool written in JAVA programming language, thus it can be ported on any operating system (i.e., UNIX, Windows CE, or Linux) without the need for source code modification.

A flood of e-mail messages from various sources such as friends or marketing companies can cause people to give up reading all of the e-mail messages. Of course, commercial e-mail checking programs provide convenient maintenance functions such as blocking address or automatically separating junk mail. A rule for categorizing junk mail might be based on e-mail addresses or titles of mail messages. As a result, important mail for a person may likely be thrown away with real junk mail. So the classification rule for the junk mail could be set up based on the contents of e-mail messages rather than sender's address or title of message. This could be implemented using eXtended Markup Language (XML). Thus, there will be a future research need for developing Mail Checker with an added function of personalization using XML.

References

Booch, G., Rumbaugh, J., and Jacobson, I. The Unified Modeling Language User Guide, Addison-Wesley, Longman Inc., Massachusetts, 1999

Constant, D., Sproull, L., and Kiesler, S. "The Kindness of Strangers: The Usefulness of Electronic Weak Ties for Technical Advice," *Organization Science* (7:2), 1996, pp.119-135

Deitel and Deitel *Java How to Program*, 3rd ed., Prentice-Hall Inc., New Jersey, 1999

Douglas, E. C. Computer Networks and Internets with Internet Applications, 3rd Ed., Prentice-Hall Inc., New Jersey, 2001, pp.455-469

JAVA 2 Complete, SYBEX Inc., Alameda, State 1999

Nemil T., and Hemrajani, A. "Introduction to the Java Mail API" Java World, 1999 (June). http://www.javaworld.com/javaworld/jw-06-1999/jw-06-javamail.html

"Java Mail Specification Version 1.2," 2000 (Dec.) http://java.sun.com/products/javamail/

Myers, J., and Rose, M. "Post Office Protocol – Version 3 ", Network Working Group, 1994 (Dec.), ftp://ftp.isi.edu/in-notes/rfc1725.txt

Savitch, W. Java: An Introduction to Computer Science and Programming, Prentice-Hall Inc., New Jersey, 1999.