MIDDLE EAST TECHNICAL UNIVERSITY

DEPARTMENT OF COMPUTER ENGINEERING

CENG491 FALL 2006

SENIOR DESIGN PROJECT

INITIAL DESIGN REPORT

MULTIWAY

by

SANZATU YAZILIM

1. INTRODUCTION

1.1 Purpose of this Document

The purpose of this document is to initiate the design specifications of the project. In this report, we intend to give our initial information about what are our solutions and how they fulfill the problem requirements. During our studies on this report, we have developed our sight to the problem and to the solution. We will present our project’s modular specification and ER and UML diagrams through which our understanding of the components of the system improves. We will present the detailed design of our system in the final design report.

1.2 Project Scope

Multiway is a server application software package by which a communication system including Hypertext Transfer Protocol, Network News Transfer Protocol, Simple Mail Transfer Protocol, Internet Message Access Protocol, File Transfer Protocol and RSS feed can be built up. This system will mainly provide the following facilities;

- ✓ Access to the system platform via HTTP and NNTP.

- ✓ Automatic email posting from system to system members.
- ✓ System members being capable of posting messages to the groups via sending email.
- ✓ RSS feed support.
- ✓ Spam filtering for received messages and received emails.
- ✓ File transfer among members.

1.3 Project Application Areas

Our project product has wide application areas. All the applications needing interaction with subscribed users are possible to be implemented by other system. Some of these areas are below:

- ✓ *Distance or Traditional Education*: The interaction between teachers and students may be provided by the newsgroups. There may be separate groups for different courses which give the students the opportunity to discuss their ideas and by our chat system which we may implement as an additional feature students can talk to each other online. Students can also upload their homework to the area created by the administrator of the newsgroup of the course.
- ✓ *Companies*: The interaction between managers and workers may be provided by the newsgroups. There may be separate groups for different departments which give the workers the opportunity to discuss the projects and exchange ideas. Workers can also upload their weekly reports to the area created by the administrator of the newsgroup of the department.
- ✓ *Online Forums*: This application can use our web module. People exchange ideas and share files on the web by subscribing to the forums formed by different newsgroups. File sharing will be managed by the administrator of each newsgroup.

1.4 Design Issues

✓ Language

We implement a number of protocols and their servers in our system. They are supposed to communicate each other to perform system tasks. We use open source servers for these tasks of the system core. Each of the servers we use is written in different programming languages. In order to integrate them into one big core, we intend to use Java as the main language, because it has packages for server implementation and for providing their communication. We may also use PHP and Perl for the web design of our system.

✓ User Interface

We make a system of news exchange between people and it will directly be used through some user interface. So, GUI is an important aspect of our project. There are different user interfaces for different user types and so easy use is one of the most important issues of the design of our system. We took this into consideration while creating user interfaces.

✓ Maintenance

We will implement a number of protocols and their servers in our system. And the system will have lots of concurrent users. So, maintenance turns out to be an important and a difficult issue. We have thought of this while creating the modules, especially the filtering module.

2. COMPONENT LEVEL DESIGN

2.1 System Modules

✓ Administrator Module:

The administrator will login to manage the system via the web interface, after the username/password verification. After the verification step, he/she could:

- create new groups and subgroups
- edit/remove existing groups and subgroups
- create/remove a user account
- create file transfer data area
- arrange user list of the system and the private groups
- arrange newsgroup chiefs
- set the duration of the posts of the group
- set the special settings differing due to public, private special groups
- perform tasks that can be done by any subscribed user.

✓ Web Module:

As our core is a web-based application, all users will be provided an interface for each type of process. Thus our web module will provide the users with an interface to login to the system for monitoring newsgroups and/or articles etc. In addition, this module will also provide the administrator with an interface to manage the system. (This part was explained in Administrator Module.) At first, the subscribed user will be asked for his/her username and password for the authentication. After the username/password verification, the user will have the right to monitor the private newsgroups and the articles. Furthermore, user could change his/her personal settings such as personal information, subscribed groups, RSS feed request or the messages that he/she wants to save. Besides the subscribed users, the unsubscribed users can only monitor and read the public groups and articles via the web page.

✓ FTP Module:

Our server is able to manage file transferring to and from the clients. There are two types of file transfer; upload and download. Using the data channel created, the server does the job requested by the client according to the commands coming through the command.

✓ RSS Module:

The requests that were sent from client are handled in “Request Handler”. Request handler determines the type of the request (NNTP/HTTP) and sends it to the related servers (NNTP/WWW). Servers send the desired files to the RSS feed. RSS feed converts it to the XML format and sends to database. Servers take the related files from the database and send the system output to the client.

✓ NNTP Module:

Our system has an NNTP module for dealing with coming NNTP requests. After authentication of the user by the system, NNTP requests are directed to this module. Coming requests are parsed to acquire <from>, <subject>, <text>, <expires> areas of requests. If the request is a read request, this module controls whether user is allowed to read articles from requested newsgroup. If the user is allowed to read requested articles, this module will find the articles in database and send them to the user via NNTP protocol. If there is a write request, the user is again controlled to be allowed to write articles to the requested newsgroup. After spam filtering on coming articles, filtered articles will be inserted to the article database of the newsgroup.

- ✓ **Receiving Email Module:**

Each member of the Multiway system has his own email account with a local host domain. Users are able to post messages to the groups by sending e –mails. When a user sends an email to the system, it is delivered to the SMTP server which afterwards transmits the e-mail to the spam filter. Then the mail is put into the IMAP server system account. Later the sender, date, to and body parts of the mail is extracted and put into the database in order to make the core access it.

- ✓ **Sending Email Module:**

Users are able to select groups from which e-mails of the posts will be sent to them. If there is a new post in these groups, e –mails of the users who have selected the option and the message content will be taken from the database. Then with these data the mail text will be formed and put into the local host accounts of the users.

2.2 Algorithmic Model (PDL)

✓ Administrator Module:

```
AdminRequest
input request
output response
       get request from client
       if(admin not verified)
            get username from admin
            get password from admin
            if(verified username/password)
                 return admin interface
            else
                 reject request
       else(admin verified and admin command taken)
            if(create new group/subgroup command)
                 get new info about group/subgroup
                 put the new group/subgroup data into database
            else if(edit/remove existing group/subgroup command)
                 trace the database for the existing group/subgroup*
                 update the existing group/subgroup*
            else if(create new user account command)
                 get the new user info.
                 add the new user account to the database
            else if(remove the user account command)
                 find the specified user account and delete it from database
```

✓ Web Module:

```
ClientHTTPRequest
input request
output response
       get request from client
       if(client not verified)
            get username from client
            get password from client
            if(verified username/password)
                 return ServerHTTPResponse
            else
                 return public articles
       else if (client verified)
```

```
            if( request for updating data)
                 get new data from client
                 return ServerHTTPResponse
            else if(request for data)
                 return ServerHTTPResponse

ServerHTTPResponse
input request
output response
       if(request for data)
            get requested data from database
            return requested data in Html format
       else
            put the new data into the database
            return response Html file
```

- ✓ **FTP Module:**

```
FTPServerRequest
input request
output none
       Loop forever
       f there is a new request
            Turn on command channel
            Check the request type
            return FTPServerResponse

FTPServerResponse
input response
output none
       If request type is upload
            Data channel is on
            Get uploaded file through the channel
            Save file in the file system
       Else if request type is download
            Data channel is on
            Take requested file from the file system
            Send the file through the channel
```

- ✓ **RSS Module:**

```
RssRequest
input request
output none
```

```
        get request from client
  determine the type pf the request
  if(nntp request)
        go to Nntp Server
        send nntp file to the rss feed
        convert nntp file to the rss file in rss feed
        RssResponse
  else if(html request)
        go to Web Server
        send html file to the rss feed
        convert html file to the rss file in rss feed
        RssResponse
   move the the rss file from rss feed to the database

RssResponse
input response
output none
  get rss file from the database
  if (nntp response)
        move the file to the nntp server
        get the system out put from nntp server
  else if
        move the file to the web server
        get the system out put from web server
  send the system out put to the client
```

✓ NNTP Module:

```
NNTPServerRequest
input request
output none
        Loop forever
        If there is a new request
              Parse request, get Request Line, Headers and Body if exists
                    If request is read request
                          Send NNTPServerReadResponse(request)
                    Else If request is write request
                          Send NNTPServerWriteResponse(request)
                    Else If request is add / remove newsgroup request
                          If user is admin
                                add / remove newsgroup
NNTPServerReadResponse
input request
output none
        If user is allowed to read requested newsgroup
              find message from database
```

```
NNTPServerWriteResponse
input request
output none
       If user is allowed to write to requested newsgroup
            filter coming message
            If spam is detected
                 throw away coming message
            Else insert new message to database
```

✓ Receiving Email Module:

```
(1)SMTP server request
input mail text
output none
Deliver mail to spam filter
Loop forever
       If there is a new  request goto (1)
            Deliver mail to user host account
            (2)IMAP server request
            Input none
            Output mail text
            Get mail sender,date,to,body parts to the
database
            Loop forever
            f there is a new request goto (2)
```

✓ Sending Email Module:

```
If  there is a new message at the subgroup table
       Get users email info related with subgroup
       Get message context
       Construct to,sender,date ,body parts of the mail
SMTP server request
Input  mail text
Output  mail forwarded
       Delivery agent request
       Input mail
       Output mail
       Loop forever
```

2.3 Sequence Diagrams

- ✓ **Administrator Module:**

- ✓ **Web Module:**

✓ **FTP Module:**

✓ **RSS Module:**

- ✓ **NNTP Module:**

- ✓ **Email Module:**

3. ARCHITECTURAL DESIGN

3.1 Data Flow Diagram

<u>Data Flow Level 0:</u>

<u>Data Flow Level 1:</u>

Data Flow Level 2:

Web Module:

News Exchange Module:

Receive Email Module:

Send Email Module:

File Transfer Module:

RSS Module:

3.2 State Transition Diagram

STD for Administrator:

STD for Newsgroup Chief:

STD for Subscribed User:

STD for HTTP Server:

STD for NNTP Server:

STD for Email Server:

STD for FTP Server:

4. DATA DESIGN

4.1 Internal Server Data Structures

Since we are planning to use open source servers for handling different types of protocols, our data objects include internal file structure of these open source servers.

- ✓ ***Courier Server: (for SMTP, IMAP and POP3 server):***

Courier implements SMTP extensions with TSL/SSL option for mailing list management and spam filtering. Courier can function as an intermediate mail relay, relaying mail between an internal LAN and the Internet, or perform final delivery to mailboxes. Furthermore, local mailboxes can be accessed via the integrated IMAP and POP3 servers which are available as separate packages. For SMTP, IMAP and POP3 server, courier defaults to storing mail in maildirs, not traditional flat file mailbox files. Actually, IMAP andPOP3 servers can be used only if mail is stored in Maildirs.

Courier can also provide mail services for regular operating system accounts. It can also provide mail services for virtual mail accounts, managed by an LDAP, MySQL, or PostgreSQL-based authentication database.

Maildir mail storage format:

Maildirs were originally implemented in the Qmail mail server, supposedly to address the inadequacies of mbox files. In maildirs, individual messages are saved in separate files, one file per message. There is a defined method for naming each file. There's a defined procedure for adding new messages to the maildir. No locking is required. Multiple processes can use maildirs at the same time. Thus maildirs require less I/O and CPU resources.

A maildir contains three subdirectories: tmp, new, and cur. These three subdirectories comprise the primary folder, where new mail is delivered by the system. Folders are additional subdirectories in the maildir whose names begin with a period: such as .Drafts or .Sent. Each folder itself contains the same three subdirectories, tmp, new, and cur, and an additional zero-length file named maildirfolder, whose purpose is to inform any mail delivery agent that it's really delivering to a folder, and that the mail delivery agent should look in the parent directory for any maildir-related information.

In courier, maildirs and maildirs folders can be created by using the maildirmake command.

Messages

Since the Courier uses maildirs to storing mail, e-mail messages are stored in separate, individual files, one e-mail message per file. The tmp subdirectory temporarily stores e-mail messages that are in the process of being delivered to this maildir. The tmp subdirectory may also store other kinds of temporary files, as long as they are created in the same way that message files are created in tmp. The new subdirectory stores messages that have been delivered to this maildir, but have not yet been seen by any mail application. The cur subdirectory stores messages that have already been seen by mail applications.

In Courier, in order to add new mail to maildirs, first a new unique filename is created, than a series of system calls, namely 'stat', 'write', 'fstat' etc., is checked.

In order to read mail from maildirs, the following steps should be followed:

-When opening a maildir or a maildir folder, read the tmp subdirectory and delete any files in there that are at least 36 hours old.

-Look for new messages in the new subdirectory. Rename new/filename, as cur/filename:2,info. Here, info represents the state of the message, and it consists of zero or more boolean flags chosen from the following: "D" - this is a 'draft' message, "R" - this message has been replied to, "S" - this message has been viewed (seen), "T" - this message has been marked to be deleted (trashed), but is not yet removed (messages are removed from maildirs simply by deleting their file), "F" - this message has been marked by the user, for some purpose. These flags must be stored in alphabetical order. New messages contain only the :2, suffix, with no flags, indicating that the messages were not seen, replied, marked, or deleted.

Actually, if the the voluntary quota protocol is used, the deliverquota command is a tiny application that delivers a single message to a maildir and hopefully it can be used as a measure of last resort. Alternatively, the libmaildir.a library can used to handle all the low-level dirty details for them.

<u>Maildir quotas</u>

Courier can manually enforce a purely voluntary quota for mail accounts that use maildirs. This quota enforcement is implemented entirely in software, and is available only when maildirs are used. This quota implementation will also work with virtual accounts. Maildir quota support is available only with userdb, LDAP, MySQL and PostgreSQL authentication back-ends. Maildir quotas are also supported by IMAP, POP3, and the webmail server. To add a maildir quota with userdb, run the following commands, for example:

```
userdb account set quota=quota
makeuserdb
```

Here, account identifies the account to which the quota applies, and quota is the quota specification.

<u>Filesystem quotas</u>

The maximum disk space quota is implemented within the operating system's filesystem support code. The operating system enforces the maximum disk space that can be used by each account. This is the only reliable quota implementation if individual accounts have login access to the mail account. Maildir quotas are implemented entirely within the maildir support code, and can easily be superceeded by anyone with login access to the mail account. Additionally, mail accounts must all be system accounts. Virtual accounts -- that share the same physical system userid -- cannot usually be support by filesystem-based quotas, because all mail accounts have the same userid and groupid in Courier.

- ✓ ***INN: (for NNTP server):***

NNTP module includes INN open source server. INN stores articles in individual text files in a directory structure. For example, article 12345 in news.software.nntp is stored as *news/software/nntp/12345* relative to the root of the article spool. Individual article data object has attributes as in the ER diagram. One of the main tasks of our core is transferring article information from file system of INN to our central database.

✓ ***Lighttp: (for HTTP server):***

Lighttpd is a fast webserver with minimal memory footprint. It is rapidly redefining efficiency of a webserver. It has effective management of the cpu-load and advanced feature set, namely FastC GI. CGI, Authentication, Output-Compression, URL-Rewriting etc. It also supports virtual hosts and virtual directory listings.

In order to support virtual host, there is a module, mod_mysql_vhost. With this module system administrator can store the path to given host's document root in a MySQL database. It is important that only one vhost module should be active at a time.

mod_auth which is a module in Lighttpd enables for authentication. Lighttpd supports both authentication method described by RFC 2617: basic and digest. Depending on the method lighttpd provides various way to store the credentials used for the authentication.

For basic authentication:

- plain
- htpasswd
- htdigest
- Idap

For digest authentication:

- plain
- htdigest

To supply a secure system, we will use digest method and htdigest for storing data in our Multiway core.

digest method:

The Digest method only transfers a hashed value over the network which performs a lot of work to harden the authentication process in insecure networks.

htdigest:

A file which contains username, realm and the md5()'ed password seperated by a colon. Each entry is terminated by a single newline. e.g.:

agent007:download area:8364d0044ef57b3defcfa141e8f77b65

4.2 ER Diagrams

- ***Entities***

✓ ***Relationships***

4.3 Database Table SQL’s

```
CREATE TABLE User(user_id INTEGER,
             username VARCHAR(20),
             passwd VARCHAR(20),
             domain_email_addr VARCHAR(100),
             real_name VARCHAR(30),
             surname VARCHAR(30),
             user_quota INTEGER,
             user_type INTEGER,
             PRIMARY KEY(user_id));


CREATE TABLE Administrator(admin_id INTEGER,
                   username VARCHAR(20),
                   passwd VARCHAR(20),
                   real_name VARCHAR(30),
                   surname VARCHAR(30),
                   user_quota INTEGER,
                   PRIMARY KEY(admin_id));


CREATE TABLE Newsgroup(group_name VARCHAR(100),
                  PRIMARY KEY(group_name));


CREATE TABLE Subgroup(subgroup_name VARCHAR(150),
                 email_address VARCHAR(200),
                 group_type INTEGER,
                 manager_id INTEGER,
                 rss_feed INTEGER,
                 ftp_directory VARCHAR(100),
                 PRIMARY KEY(subgroup_name));


CREATE TABLE Has_subgroup(group_name VARCHAR(100),
                  subgroup_name VARCHAR(150),
                       PRIMARY KEY(group_name, subgroup_name),
                  FOREIGN KEY (group_name) REFERENCES Newsgroup,
                  FOREIGN KEY (subgroup_name) REFERENCES
Subgroup);


CREATE TABLE Post(message_id INTEGER,
              subject VARCHAR(200),
              sender_id INTEGER,
              send_date DATE,
              expires INTEGER,
              message_path TEXT,
              message_text TEXT,
              PRIMARY KEY(message_id));

CREATE TABLE Has_post(subgroup_name VARCHAR(150),
                 message_id INTEGER,
                 PRIMARY KEY(subgroup_name, message_id),
                 FOREIGN KEY (subgroup_name) REFERENCES Subgroup,
                 FOREIGN KEY (message_id) REFERENCES Post);



CREATE TABLE Reads_writes(user_id INTEGER,
                  subgroup_name VARCHAR(150),
                  FOREIGN KEY (user_id) REFERENCES User,
                  FOREIGN KEY (subgroup_name) REFERENCES
Subgroup);



CREATE TABLE File(file_id INTEGER,
              file_name VARCHAR(100),
              file_size INTEGER,
              file_type DATE,
              PRIMARY KEY(file_id));



CREATE TABLE Uploads_downloads(user_id INTEGER,
                       file_id VARCHAR(150),
                       FOREIGN KEY (user_id) REFERENCES User,
                       FOREIGN KEY (file_id) REFERENCES Files);
```

4.4 Data Dictionary

Name: User
Alias: -
Where/How used: The people that will use the system.
Description: Every actor using the system is defined to be a user.

Name: username
Alias: -
Where/How used: While logging into system
Description: Username is a unique name given to the users for the system security. The users will enter their username together with their passwords to log into system.

Name: passwd
Alias: password
Where/How used: While logging into system
Description: The password is to secure the system. Unauthorized users cannot log into the system. The user's password is converted to MD5 hash and checked with the one in the database.

Name: user_type
Alias: -
Where/How used: -
Description: User type can be a subscribed/unsubscripted user, the chief of a subgroup or administrator.

Name: domain email addr
Alias: -
Where/How used: User will send mail to newsgroups from this email address.
Description: When a user is added to the system, an unique domain email address is given his/her.

Name: real_name
Alias: -
Where/How used: -
Description: The real name of the users will be recorded in database.

Name: surname
Alias: -
Where/How used: -
Description: The real surname of the users will be recorded in database.

Name: user_quota
Alias: -

Where/How used: User can keep the messages that he/she wants in his/her account,
until their quota is up to limit size.
Description: User quota specifies the size of user's account.

Name: Admin
Alias: Administrator
Where/How used: -
Description: Admin is a special type of user who has all privileges. He/she will be able to access the admin interface after validation of username/password. From this interface, he/she can:

- create new groups and subgroups
- edit/remove existing groups and subgroups
- create/remove a user account
- create file transfer data area
- arrange user list of the system and the private groups
- arrange newsgroup chiefs
- set the duration of the posts of the group
- set the special settings differing due to public, private special groups
- perform tasks that can be done by any subscribed user.

Name: Newsgroup
Alias: -
Where/How used: It will be used for categorizing the groups.
Description: Newsgroup is the upper group that is opened by admin. It has more than one subgroups. Each newsgroup will have unique name.

Name: group_name
Alias: -
Where/How used: Group name is used for specifying the email address of the subgroups concatenating with subgroup_name.
Description: It specifies the name of the newsgroups.

Name: Subgroup
Alias: -
Where/How used: It is the sub group of newsgroup that will be used by users.
Description: Subgroup that is the sub group of newsgroup has an unique name.

Each newsgroup could have more than one subgroups.

Name: email_address
Alias: -
Where/How used: When the user sends mail to newsgroups, the receiver will be the unique email address of subgroup.
Description: Email address of the subgroups will be formed by concatenating the group name of the newsgroup and the subgroup name. Thus it will be unique for each subgroups.

Name: subgroup_name
Alias: -
Where/How used: Subgroup name is used for specifying the email address of the subgroups concatenating with group_name.
Description: It specifies the name of the subgroups.

Name: manager_id
Alias: -
Where/How used: When admin create a private subgroup, he/she will specify a subgroup chief that has unique manager id.
Description: It refers to unique user_id of the user who will manage the specified subgroup.

Name: rss_feed
Alias: -
Where/How used: -
Description: Rss_feed specifies the If the rss_feed is equal to one than the RSS
feed

Name: ftp_directory
Alias: -
Where/How used: When admin create file transfer data area for a subgroup, the name of the directory of that data area is kept in database with the specified name, ftp_directory.
Description:

Name: group_type
Alias: -
Where/How used: -

Description: All subgroup will be private, protected or public to users. That means: if subgroup is public, all users, including unsubscripted users, can monitor/read it, if it is protected, only subscribed users can monitor/read/write it, and if it is private than only the subscribed users who are specified by subgroup chief.

Name: Post
Alias: Message, mail
Where/How used: When the user post message or send mail to system, or when the system send mail to users.
Description: Post is the message/mail that is sent between the users and the system.

Name: message_id
Alias: -
Where/How used: It is used for keeping messages in database sequentially.
Description: All message has unique message_id.

Name: subject
Alias: -
Where/How used: When user sends mail or posts message to system, he/she will write a subject.
Description: -

Name: send_date
Alias: -
Where/How used: The message/mail will be kept in database and shown to users in a period of expires beginning on the send_date.
Description: The date of the messages/mail are kept in database.

Name: message_text
Alias: -
Where/How used: -
Description: The users messages/mail sent to system are kept in message_text.

Name: sender_id
Alias: -
Where/How used: While showing messages in newsgroups, the sender will be shown. To show the sender name, the name of the sender will be extracted

by checking sender _id with user_id.
Description: Sender id is actually the user id of the sender.

Name: expires
Alias: -
Where/How used: All the messages in the system will be shown in a period of expires beginning on the send date of the them.
Description: Expires is a period of time which is indicated by days.

Name: File
Alias: -
Where/How used: When user uploads or downloads file from the system.
Description: Any kind of file used in the system.

Name: file_id
Alias: -
Where/How used: -
Description: The unique id for each file in the system.

Name: file_name
Alias: -
Where/How used: -
Description: A unique name given to each file in the system.

Name: file_type
Alias: -
Where/How used: While storing file into database.
Description: Each file type is kept in database.

Name: file_size
Alias: -
Where/How used: -
Description: The size of the file is kept in database.

Name: Keeps_post
Alias: -

Where/How used: In order to figure out the messages that user wants to keep in his/her account.
Description: Keeps_post is a relation between the User and Post.

Name: Reads_writes
Alias: -
Where/How used: When the user reads or post messages to the newsgroup.
Description: Reads_writes is a relation between the User and Post.

Name: Creates_subgroup
Alias: -
Where/How used: When chief user(also named subgroup chief) creates subgroups.
Description: Creates_subgroup is a relation between the User and Subgroup.

Name: Creates_newsgroup
Alias: -
Where/How used: When administrator creates new newsgroup.
Description: Creates_newsgroup is a relation between Administrator and Newsgroup.

Name: Has_post
Alias: -
Where/How used: When monitoring the messages that posted to subgroups.
Description: Has_subgroup is a relation between Post and Subgroup.

Name: Has_subgroup
Alias: -
Where/How used: To figure out which subgroups belongs to which newsgroup.
Description: Has_subgroup is a relation between Subgroup and Newsgroup.

Name: Download_upload
Alias: -
Where/How used: When the user upload/download files from system.
Description: Download_upload is a relation between User and File.

5. INTERFACE DESIGN

✓ *Login window:*

On the left part of the login window there is a label that contain the names of the groups with their subgroups and also a checkboxes near the names of all subgroups to show if the related subgroup support Rss feed or not. The purpose of showing the group names on the login window is allowing some groups to be read without login.

On the right part of the login window there are text edit boxes that have to be filled with correct information to login. These informations are “User Name”,”Password”,”email adress” and also a two text button for spam control.

On the buttom of the right part of the login window there is a button named “LOGIN”.To log the system the user have to click that button after filling the wanted information.

✓ *User main window:*

On the top of that window there are six bottom named “HOME”, ” GROUPS”, “MAIL”, “PERSONAL ACCOUNT”,”FILE DOWNLOAD”,”LOGOUT”. The purpose of that buttons are as guested through their names is going to home window,going two group window,going to mail window,going to personal account window,going to file download window and logout from the system.

- ✓ ***Newsgroup window for login User***

On the left part of the main window there is a label that contain the names of the groups with their subgroups and also a check boxes near the names of all subgroups to show if the related subgroup support Rss feed or not.

On right part of the main window:

On the top of the that there are information for the user who is logged (user_ıd,name,surname,department).Moreover the group name that is currently active can also be seen on the top. Lastly on the top there is a button named “WRITE” . When user push that button window for sending email will be open.

On the main of that part news of the selected subgroup is arranged. User can see the date,subject,author of a new in the main part.

On right panel there are also check boxes. User click the check boxes that are near the news he/she want to save.

On the bottom of the window there is a label that user can read the news, he/she click on the main part of the above right window. On that label there are also two more buttons names “REPLY”,”CANCEL”. By the help of this buttons user can reply to a news or cancel the news that he/she send before and now wanted to delete.

✓ *Mail send window for login User*

On the top of the window there are three boxes named “HOME”,”Inbox”,”Compose Mail”, ”Sent”.By clicking the first window user can go to home window,by clicking the second window he/she can see the inbox, by clicking the compose window he/she can compose an email to send,by clicking the last window he/she can send the mail he/she compose. On the next of that buttons there is label that gives information about the quota that user used in her/his account.

On the main part of that window user can see the mails that are sent to her/him and also check boxes near the window. If user click a check box near a mail and click the delete button on the top the mail is deleted.

- ✓ ***File upload/download window***

1) In that window user choose the group he/she wanted to upload/download

file.

2) In that window on the top user can see the group name that the file will be uploaded/downloaded. On the below of that there are a list of file that are available to download. If user click the check box near one of the file in that list and click the “Download” button in the window he/she downloaded the file. On the below of the part for downloading file there is the part for uploading file. In that part user see the title and deadlines of files that can be loaded and select the part that he/she wanted to upload the file according the the title of the file. When user click the select button in there the third window will be opened.

3) In that window user see the title and the deadline max size for the file that will be uploaded and the name of the chief. Also user will browse the file that he/she will be upload and that click the upload button to upload the file.

✓ *Newsgroup Chief Main Window*

On the top of that window there are six bottom named “HOME”, ” GROUPS”, “MAIL”, “PERSONAL ACCOUNT”,”FILE DOWNLOAD”,”LOGOUT”. The purpose of that buttons are as guested through their names is going to home window,going two group window,going to mail window,going to personal account window,going to file download window and logout from the system.

On the right of thye main part of the window there are a button for selecting the group. On the left there edit text boxes("File Title","Deadline" that are have to filled correctly to arrange file operations. Moreover there is a button named "Let Upload" . After chief click that button in the database a new part is produces for file transfer. Chief can also browse the file he/she wanted to upload and click the upload button near it.

Main window of Multiway admin interfaces contains "Create a Group","Create a Subgroup","Add User","Remove User","Edit User","Select Different Server" and "Edit Servers" buttons in addition to general user fields. When administrator pushes these buttons new windows related to these actions are popped up.

✓ *User Personal Account Window*

On the top of the that there are information for the user who is logged (user_ıd,name,surname,department).And also a button(HOME) that takes the user to the home window. On the main part of the window there are three buttons named "Change

Password” that is used for changing the password,”View Saved Messages” that is used for viewing the messages in the user account,”Login History” that is used to seing the last login information of the user.

✓ *Admin Main Window*

Main window of Multiway admin interfaces contains “Create a Group”,”Create a Subgroup”,”Add User”,”Remove User”,”Edit User”,”Select Different Server” and “Edit Servers” buttons in addition to general user fields.When administrator pushes these buttons new windows related to these actions are popped up.

✓ ***Create Subgroup Window***

When administrator pushes “Create a Subgroup” button from the main window the window above is popped .Admin selects the group to which the new subgroup will belong to ,enters the subgroup name to “Group Name” field,chooses RSS feed option and the duration time for the posts of the new newsgroup.As mentioned before a subgroup may be “Public”,”Protected” or “Private”.Public groups can be readable by anyone but unregistered people can not write to these groups.Protected groups can be read/written to only by registered users.Private groups belong only to the members who are specially defined by the administrator.Admin enters the user id's of the members of the private groups by uploading a special formatted file,so that only members with these user-id's can read and write to these private groups.

✓ *Add/Remove User Window*

People are allowed to use the system only if they are added to the system by the administrator.When admin pushes “Add User” button from the main window ,the new window shown above appears. After entering the name and surname information,admin pushes “Create Password “and “Create User-Id “ buttons so that system creates unique id and password for the new member.Afterwards email account of the user is created with the e-mail address beginning with user id followed by local domain name.

Administrator makes the members unsubscribe from the system by the window created when he pushes “Remove User” from the main window.After entering the user-id,name and surname information,user is no longer a member of the system after pushing the “Remove User” button.

✓ *Edit Server Window*

Administrator is able to configure SMTP and NNTP servers by his interface.When he pushes “Edit Servers” button from the main interface window,the window above is pushed.By the 6 buttons shown above ,admin can configure the properties written on the buttons by the windows popped up when each button is pressed.

✓ *Select Different Server Window*

Multiway may be installed on the server machine or on a remote machine.By pushing select different server button, administrator can configure the settings for a remote virtual server.

6. PROJECT REQUIREMENTS

6.1 Functional Requirements

- ***Create a new newsgroup on the system:***

This function will be used by the administrator in our system. He may create a new newsgroup on the system by his own will or on desire of the newsgroup chiefs. A name for the group is required when it is created and also user permissions of the group should be determined. Depending on its permissions, a newsgroup may be public (read or written by all users), private (read or written by subscribed users only) or private private (read or written by some subscribed users). The administrator should decide on this functionality when he creates the newsgroup. If the created newsgroup is private private, its subscribed users and its chief should be determined. The administrator should also register if RSS functionality would be used for the created newsgroup and how long the posts of the group would be kept in the system database.

- ***Edit/Remove a newsgroup from the system:***

The functionalities of some newsgroup would be edited a while after it is created. The newsgroup would also be removed. These functionalities are also managed by the administrator.

- ***Create/remove a user account:***

There are some subscribed users of the system which have personal accounts in the system database. The user accounts are created by the administrator on demand. Subscribed users can read and write on private groups using their accounts. The user accounts are also removed by the administrator when it is desired.

- ***Create file transfer data area:***

There is an FTP extension of our system, but in addition to it, the users will send files to the system in some special times. For this purpose, the administrator is responsible for creating some area on the database on desire of the newsgroup chiefs and subscribed users of the newsgroup whose chief has desired it, will upload their files to that created area.

- ***Secure login:***

The subscribed users of the system, including the administrator and newsgroup chiefs should login the system to do actions. An authentication check will exist in the system to maintain the security. Username and password of the users will be checked to match with the ones on the database; if they match, the user will be allowed to login.

- ***Update user information:***

There will be the possibility for the subscribed users to update their personal information. This is also valid for the administrator and the newsgroup chiefs. The administrator will also be able to change user information and accounts on demand, then inform the user of this.

- ***Upload/Download files:***

The subscribed users will be able to upload or download files by using FTP URL. That is, file transfer between subscribed users (including newsgroup chiefs) will be possible using FTP. This will be valid for all the times; there will also be a file transfer channel between subscribed users and chief of a newsgroup where the chief will be the receiver and other users will be the sender. A temporary area on the database will be created by the administrator for this purpose on the desire of the chiefs.

- ***RSS Functionality:***

The administrator will decide on if a newsgroup would have an RSS Service or not. The required newsgroups will have the RSS XML files to be sent to users in some time intervals. The users will be able to choose some newsgroups to get posts from and read them by using their RSS readers.

6.2 Non-Functional Requirements

- ***Web Interface:***

The Web interface will be for all public and private users and will use HTTP protocol. Users will be able to reach all functionalities of the system by using Internet. The web interface will certainly be user-friendly and easy to use because all the users will access the system by Internet.

- ***Security:***

Because there will be lots of ways to access our system, the security gains much importance. The system will use SSL with HTTPS and SFTP to maintain the security. A Proxy Server will also be provided to the system to be both fast and secure. Because Internet will be the most used way to use our system and it is also used by unsubscribed users, it is more necessary to maintain the security on Internet.

- ***Spam Filtering:***

Our system will send emails to and receive emails from the users. At this point spam filtering will be an important issue. The system will check for spam mails and filter them for the goodness of the subscribed users and the system.

- ***Platform Independence:***

Although the servers of our system will use Unix as the platform of development, the users will be able to access the system on every platform such as Windows, Linux, etc.

- ***Performance:***

There are many user groups and will be accordingly a high number of users accessing the system at the same time. So, the system will be affected at a minimum and have a high speed performance all the time.

- ***Reliability:***

The system should be as bug free as possible. All sub components should work asynchronously, so that any delay caused by one of the components should not block other components work on its own.

6.3 Minimal Hardware Requirements

Minimal hardware requirements for our project are: A PC with the following configuration will be needed:

>>***Development***	>>***End User***
_ Intel Pentium IV 2 GHz	_ Intel Pentium IV 1 GHz
_ 1GB DDR RAM	_ 512 MB DDR RAM
_ 100GB Hard Disk Space	_ 40GB Hard Disk Space

_ Local or Wide Area Network _ Internet Connection

6.4 Minimal Software Requirements

Software requirements for the project are divided into categories: Development and End-user. For the Development part, whole the requirements of the open source servers that could be used in our project core are also considered.

- ***Development***

_ Linux Distribution such as Ubuntu 6.0 or OpenBSD 2.8 and up or FreeBSD 2.2.x and up etc.

_ C++ compiler (such as egcs, g++)

_ GNU make

_ Either the GDBM or Berkeley DB library.

_ Perl 5.

_ Python 1.5.2 or higher

_ An implementation of yacc

_ File system supporting FIFOs

_ File system domain sockets as a must.

- ***End user***

_ Recent Linux Distribution such as Ubuntu 6.0, or Windows XP (or newer) or Mac OS

_ Web Browser (Mozilla Firefox, Microsoft Internet Explorer etc) or News Reader (Mozilla Thunderbird, OutlookExpress etc)

_ Any RSS reader

7. SYNTAX SPECIFICATIONS

In our MULTIWAY project, we use more than one language. But as it is explained in the introduction part, we use mainly Java, which is an object-oriented language. And as a part of standardization process for rapid development of our project, we have decided to develop special naming conventions. In addition, we have decided the syntax of descriptive comments that will be used for understandability and maintainability of our product. The details of the specifications are described below.

We have decided that all names should be comprehensible. For names that are composed of more than one word, lower case/upper case characters will be used to distinguish between consecutive words

.

✓ ***Naming the Classes:***

All classes will have names beginning with a capital letter. The classes with more than one word will have a capital letter at the beginning of each word to make it easy to understand the functionality of the class.

✓ ***Naming the Class Attributes:***

Attributes will begin with a lower case letter. In case there are more words, they will be distinguished by underscores. "user_id" "subgroup_name" are valid class attribute examples.

✓ ***Naming the Class Methods:***

Methods will begin with a lower case letter. In case there are more words in the method name, these words will be distinguished by capital letters at the beginning of each word. "getUserId" and "addNewsgroup" are valid method names.

✓ ***Naming the Database Table:***

Names of the tables in the database will begin with capital letters and will continue with a capital letter for each consecutive word similar to the class conventions. Attributes of the tables will follow the naming convention for the class attributes; that is, will begin with lower case letters and continue with underscore before each new word.

✓ ***Naming the Files:***

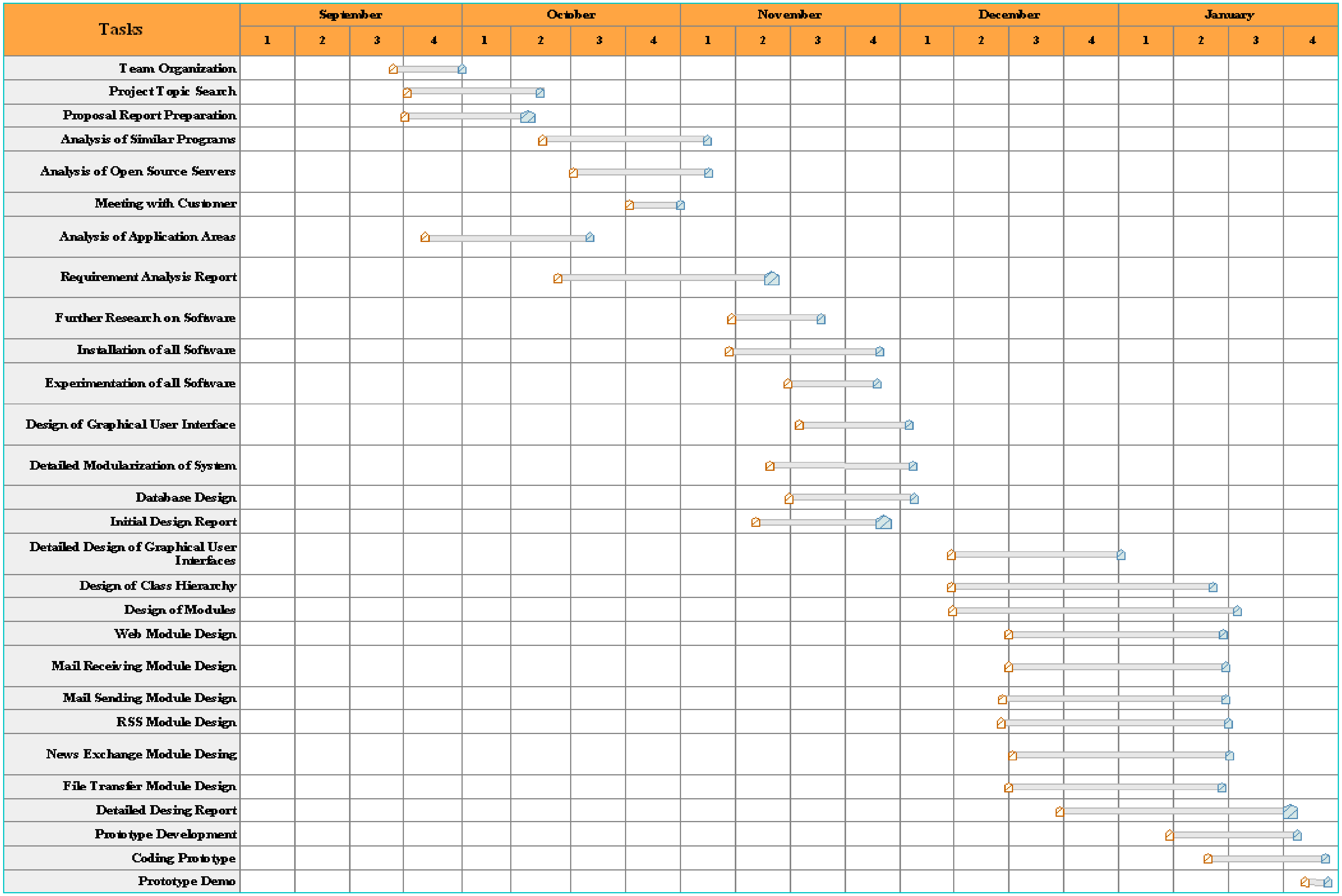
Files that include the source code for a class will be named as the following respectively:

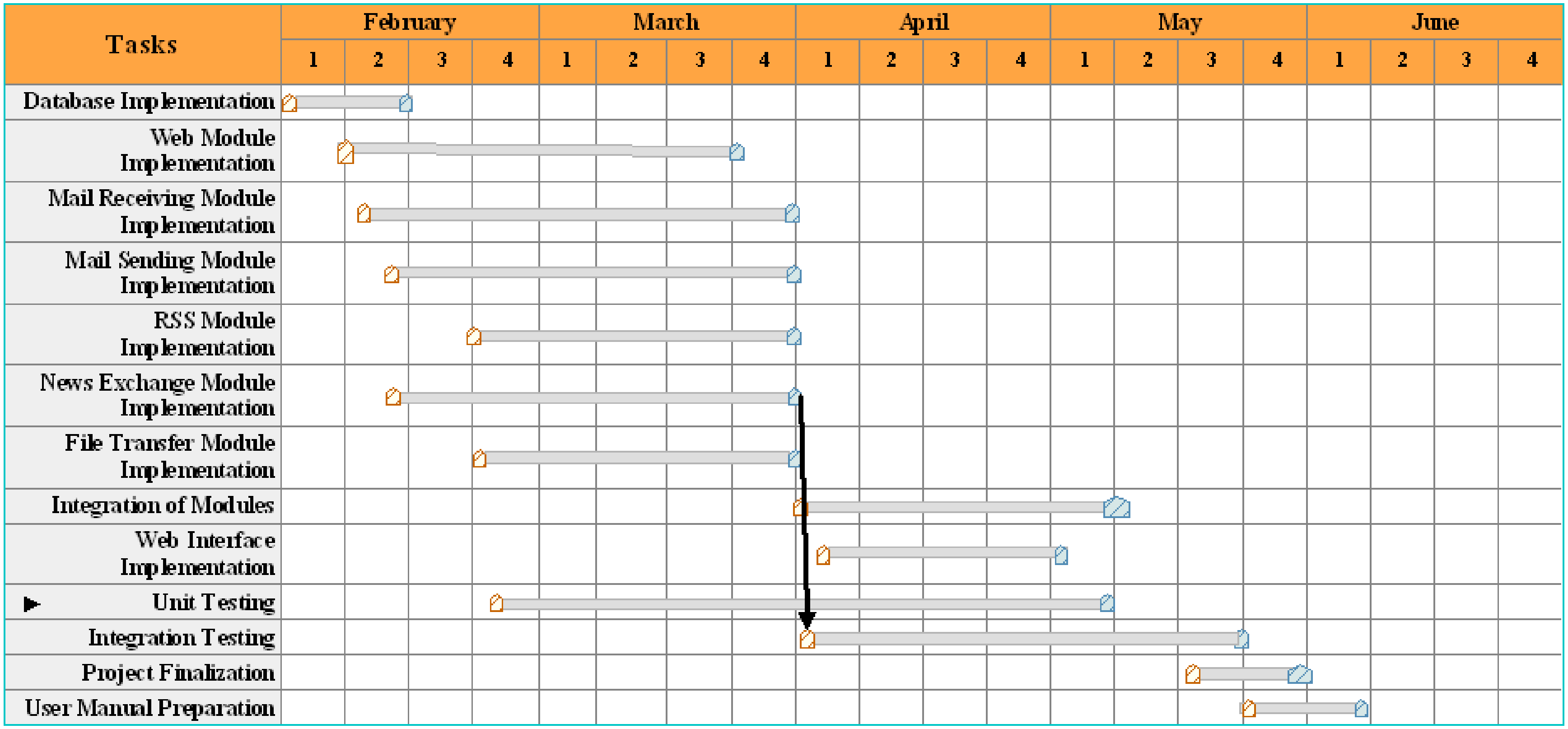
<class_name><.java>

- ✓ ***Naming the Global and Local Variables:***

We will try to avoid using global variables as an appropriate software engineering principle. However, in case of any necessity, global variables will be prefixed with "g_", since usage of global variables significantly decrease the understandability of the source code. For local variables, similar to global variables convention, the variables will be prefixed with "l_".

8.PROJECT SCHEDULE (Gannt Chart)


Tasks
September
October
November
December
January
1
2
3
4
Team Organization
Project Topic Search
Proposal Report Preparation
Analysis of Similar Programs
Analysis of Open Source Servers
Meeting with Customer
Analysis of Application Areas
Requirement Analysis Report
Further Research on Software
Installation of all Software
Experimentation of all Software
Design of Graphical User Interface
Detailed Modularization of System
Database Design
Initial Design Report
Detailed Design of Graphical User Interfaces
Design of Class Hierarchy
Design of Modules
Web Module Design
Mail Receiving Module Design
Mail Sending Module Design
RSS Module Design
News Exchange Module Desing
File Transfer Module Design
Detailed Desing Report
Prototype Development
Coding Prototype
Prototype Demo


Tasks
February
March
April
May
June
1
2
3
4
Database Implementation
Web Module Implementation
Mail Receiving Module Implementation
Mail Sending Module Implementation
RSS Module Implementation
News Exchange Module Implementation
File Transfer Module Implementation
Integration of Modules
Web Interface Implementation
Unit Testing
Integration Testing
Project Finalization
User Manual Preparation