Middle East Technical University

Department of Computer Engineering

`A Unified News Exchange Server `

Requirement Analysis Report

Goncagül DEMİRDİZEN

Hilal KARAMAN

Ali Anıl SINACI

Ferhat ŞAHİNKAYA

"NewsAgent"

by

i$T€ Yazılım

Fall, 2006

1 INTRODUCTION

1.1 MOTIVATION

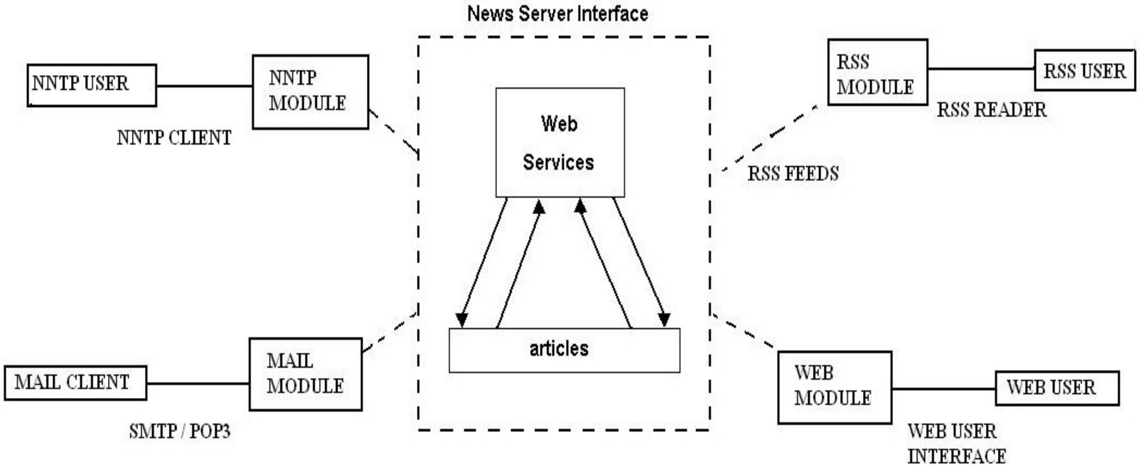
Communication has always been a significant aspect in human beings' lives. As the time passes and technology evolves, it appears with different usages and new techniques are discovered for serving communication. Accordingly, after Internet has started to be used widely, communication became one of the most important usage areas of it, especially electronic mails and online chat. Nowadays, most people use mailing lists, newsgroups or web forums for communication and reaching data about a specific issue. Definitely, these ways are more practical for now, when compared with searching whole Internet for a specific data.

1.2 PROJECT TITLE

Our project title is *NewsAgent*.

1.3 PROJECT DEFINITION

As mentioned in Motivation part, communication methods between people are changing gradually, as people discover more appropriate methods for dealing with data. Handling different access methods to data is very significant for a news server. In fact, that is the reason for developing *NewsAgent*. Following figure shows a general view of *NewsAgent*.



NewsAgent will provide users to reach data through web, tin, e-mail and news clients or via e-mail and RSS options will provide user to reach data in a fast and consistent manner. Furthermore, we can say that when *NewsAgent* takes its place in the market, users will feel the comfortable way of reaching data from different platforms.

1.4 PROJECT SCOPE

NewsAgent will contain several components, each of which will address different methods for communication. Each component will provide a different platform for communication and we can differ each user by the component that he/she used. For this reason, *NewsAgent* users can be named as NNTP user, RSS user, Web user, Mail user and administrator. Here are some general features that will be in *NewsAgent*:

- Administrator will be able to use the full power of *NewsAgent*. Administrators will be people who are responsible from the management of newsgroups. Creating new newsgroups or handling of undesirable articles in any of the newsgroups will be in the scope of his/her responsibilities.

- Web users will be able to access newsgroups and articles through a graphical interface Web user will be able to post a new article to a newsgroup or open a new thread in a newsgroup. Web component will also provide management facilities for each user and a user-friendly interface will provide user to reach data, quickly.

- NNTP users will be able to access newsgroups through tin or NNTP clients, like Mozilla, Thunderbird or Microsoft Outlook Express.

- RSS users will be able to receive feeds from newsgroups according to their wishes. By this way, as feeds provided by *NewsAgent* are updated, RSS users will be able to access new data.

- Mail users will be able to receive mails from different newsgroups according to their wishes. Of course, mail users will be able to send posts to newsgroups as a new thread or as a follow-up.

- *NewsAgent* will contain several user groups and each user group will have different access rights. Authentication will specify access rights of each user and user will be able to access different newsgroups according to their rights and newsgroups that they are subscribed. In addition to user groups, also there will be a general access right which will not need authentication and user will be able to access some subset of newsgroups which is specified by the system administrator.

2 TEAM ORGANIZATION

2.1 TEAM STRUCTURE

We decided our team structure to be Controlled Decentralized (CD). In Controlled Decentralized team structure, the team has a team leader who coordinates the team. Moreover, the team leader assigns tasks to group members in the team and each person is responsible for some subtasks. The team takes decisions together and communication between the team members is very important. Therefore we thought that a Controlled Decentralized team structure is the most suitable one for our team.

2.2 MEMBER ROLES

The roles have been distributed among the team members as follows:

i$T€ Team Members	**Member Roles**
Ali Anıl Sınacı	Team Leader, Initiator
Goncagül Demirdizen	Gatekeeper, Optimist
Ferhat Şahinkaya	Devil's Advocate, Timekeeper
Hilal Karaman	Recorder, Summarizer

2.3 PROCESS MODEL

Since we progress in the project step by step analysis, initial design, detailed design, prototype preparation, implementation, testing and maintenance, we considered that linear sequential model (Waterfall Model) is the most appropriate process model for our project. In linear sequential model, the phases are followed in a manner that when one phase is finished the next step starts. When all the requirements are specified and understood, the design step starts and according to the requirements the system is designed and after the design process the implementation of all components of the system are accomplished and the life cycle of the process moves to the testing and the faults of the earlier phases are removed here.

3 MARKET RESEARCH

3.1 LITERATURE SURVEY

Market research is one of the most important parts of the requirement analysis in the sense that it provides us to have general information on the similar systems and helps us to determine the requirements in appropriate way. With the help of market research, we had the chance of examining the similar projects to reconsider the features of *NewsAgent* and we had specified the features that users expect from such a system. For this purpose, we have examined four different projects that are similar to *NewsAgent* namely News Cache, MPNews, INN, DNews. Each of these projects contains different aspects that we have to consider in the development of *NewsAgent*. The followings are the descriptions and features of these projects.

News Cache [1]

- News Cache consists three types of newsgroups which differ from each other according to their post and read rights.

    - For reading and posting allowed newsgroups, everyone is allowed to read the articles in these newsgroups and post articles to them.
    - For moderated newsgroups, articles can be posted to newsgroups after the verification of the system moderator. Moderator verifies articles according to their contents. However, reading articles does not require such verification. Everyone can read articles in moderated newsgroups.
    - For read-only newsgroups, everyone can read articles in these newsgroups however, only users that are authorized can post articles to these newsgroups.

- News Cache developers decided to implement their own database management system, which depends on a file system, after the consideration of advantages and disadvantages of using files or tables for the storage of articles and newsgroups. Here are reasons of News Cache developers' choice:
    - Since most of the articles will have a small size about 1-4 KB and disks are allocated block by block (typically having 512 – 4096 bytes), there will be waste of disk space about 10% - 25%.
    - Since most file systems have a limit on the number of files that can be created, saving all articles as a file will cause problem of handling and storing all articles in newsgroups.
- News Cache stores the all newsgroups in their Active Database. The type of the newsgroup, the article numbers of the first and last articles in the newsgroup are stored.
- News Cache also holds an Overview Database which stores the summaries of the articles which consist of date, author, and header, number of lines and article id of the

article. Overview Database provides readers’ fast access since this summary information is retrieved from overview database and all articles are not retrieved for this purpose.

- For storing the articles News Cache follows such a way that each newsgroup has its own directory and the articles of that newsgroup are stored under the newsgroup directory in the article's own file. The file is named related with the article number.
- News Cache keeps track of a cache mechanism and articles which are requested very often are stored in the cache and when a request comes, first of all the cache is controlled and if the article is there, it is retrieved from the cache.

MPNews [2]

- MPNews provides NNTP access which means that people who use newsreader programs can access all the newsgroups of the MPNews .
- MPNews also provides a web forum option which means that the user will be able to access the articles via Web. Moreover, the user can reach other users' information and search for the articles.
- MPNews provides RSS/Atom feeds. Users can subscribe to an RSS or Atom feed and the messages in these feeds are sent to the user. For every newsgroup, a different feed is kept and updated according to the new articles coming to the newsgroup.
- MPNews provides a mailing list option in order to read the messages in a newsgroup and post messages using e-mail.
- MPNews has some restriction on the access of the newsgroups. Some newsgroups are read-only and some newsgroups are hidden from the users who are not authenticated. Users belong to some user groups which have different access rights on the newsgroups.
- MPNews also filters the messages which do not suit the content of the newsgroups.
- The user can cancel the message that he/she has sent before. In case of a such a request from a user, it is controlled whether that user has sent the message or not and if the user is the author of the message it can be cancelled. Administrators have the right of cancelling messages of other users. Moreover, messages can be deleted or searched in a newsgroup.

- MPNews deletes the old messages periodically. However, it applies different deleting policies to the newsgroups. For example the messages of the important groups stored longer.
- MPNews does not have a limit on the number of the newsgroups.

INN [3]

- INN stores each article into its own individual file in a subdirectory named related the newsgroup name.
- If an article is posted to more than one newsgroup, one way to handle this situation is to create multiple links to the file. Another approach is to copy the file and store the file for every newsgroup that it was sent.
- The message_id of the article and the location where the article is stored in a history database.
- INN supports a filtering mechanism and the administrator has the right to reject the articles which are not suitable.
- Additional packages, namely innfeed and innd, have been added to INN to support feed articles out to other servers.

DNEWS

- In DNEWS there exists header-only sucking option and the article bodies are retrieved when the user selects the article.
- DNEWS provides a dynamic sucking feed option which selects and maintains the newsgroups that the users use. Newsgroups that users read are sucked and as a result 1-10 % of disk space and network bandwidth are used and articles can be stored longer.
- DNEWS has also an expiration strategy and DNEWS deletes more articles as the disk space begins to be full.
- DNEWS provides a news to web option and users can retrieve and post articles to the newsgroups using web browsers.
- DNEWS provides a news to mail option for the users.

- DNEWS also applies a filtering mechanism in order not to accept all the coming articles.

3.2 INTERVIEW AND QUESTIONNAIRE

3.2.1 Interview with Ahmet Saçan

In order to specify our system requirements precisely, we needed to talk with someone who is experienced in such a field. For this purpose, we decided to interview with Ahmet Saçan who had dealed with a similar news server project before. When we told him our intend, he had a positive attitude towards us and we decided to meet that afternoon. Before going to the meeting, we prepared some questions and the meeting proceeded according to our questions. Below, we explained the subjects which we have discussed:

- We asked the general structure of our department's news server. At first, we were thinking that all components of the server was his own work. But we learned that he used a server called INN and integrated it with COW system.
- The most important subject we wanted to clarify was the storage of articles in our system. We couldn't decide whether to use a file system of our own or not. He explained us the way that INN stores data. It uses a file system and puts each article in file in a separate folder. It also manages the indexing and file locking mechanisms on its own. But Ahmet Saçan told that using a database will be easier for us, since this project is not based on file management.
- We told him our intend to archive the articles periodically into files. We thought that it will be useful to store older articles in files and new ones in the database. He gave us the idea of storing both of them in database using 2 different databases, since database is faster than using a file system.
- For database management, we had emphasized on PostgreSQL and MySQL. Ahmet Saçan said that it doesn't make much difference to use any of them in our project. He said that they both have advantages and disadvantages. For example, PostgreSQL is more stable and join operations are better in PostgreSQL. But MySQL is easier to use.

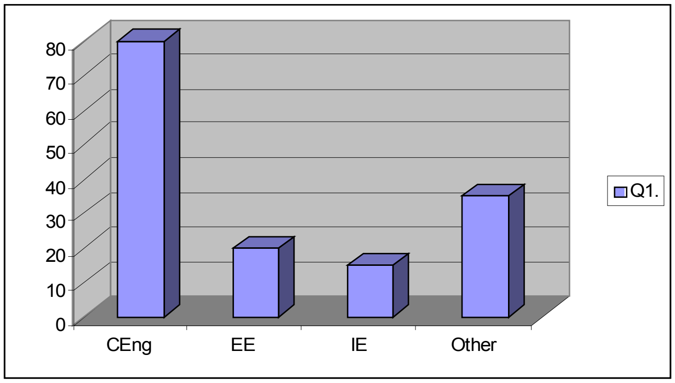
Results of the Interview

After the interview, we have found answers to most of our questions and our design decisions became clearer. We decided to store our articles in database and for archiving old articles we will make use of another database in stead of the file system. Moreover, we will use PostgreSQL for database management. We conclude that it will be more advantageous for our system requirements.

3.2.2 Questionnaire

Questionnaire is one of the most significant analyses for specifying user needs on a system that will be developed. Since questionnaire involves ideas of large number of people, analyzing these data will provide us to consider our previous decisions again, according to user needs.

1. Hangi bölüm öğrencisisiniz / mezun musunuz?

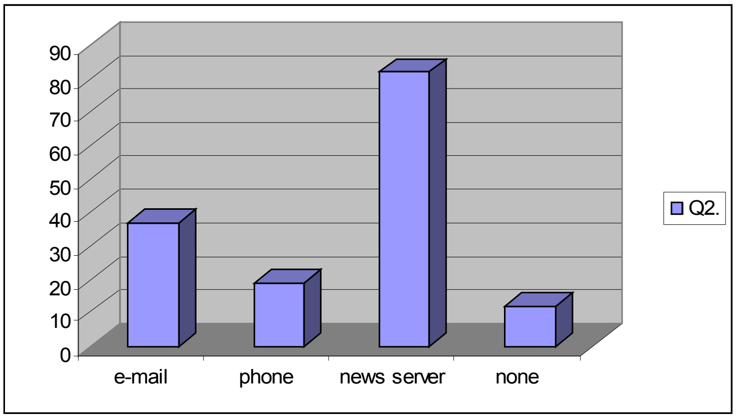


We have asked this question for specifying possible users of *NewsAgent*. Since CEng, EE and IE students or graduates of METU are the most familiar group with newsgroups, it is reasonable to conduct our questionnaire on them.

As above bar graph shows, a large percentage of our survey consists of CEng students our graduates, which is again reasonable to get really realistic and applicable features on *NewsAgent*.

2. Çalışıyorsanız iş yerinizdeki, öğrenci iseniz okulunuzdaki insanlarla nasıl haberleşiyorsunuz?

□ e-mail □ Telefon □ Haber Sunucusu □ Hiçbiri

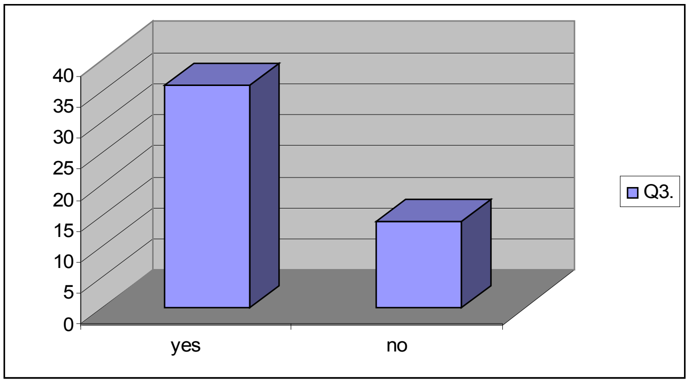


As above graphical distribution shows most of the people under survey, they use news servers for communication at school and work. In fact, we could guess the possible results before conducting questionnaire, since we have conducted our questionnaire on mostly computer engineering students and graduates who use news server of our department.

But if we deal with percentages of other methods used for communication, people mostly use e-mails or mail groups of their working places. There are some disadvantages of using this method. For instance, a person who wants to send e-mails to large number of people, will be in trouble. Instead of mailing to large number of people, posting an article to a newsgroup will satisfy his/her needs. Furthermore, using phones for communication is not practical at all.

3. İletişiminizi sağlamak amacıyla bir haber sunucusu kullanmak ister misiniz?

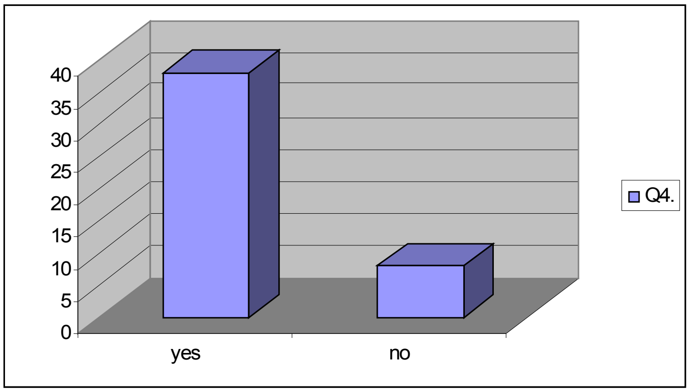
□ Evet. □ Hayır.



Since most of the people were aware of what a newsgroup is, they answered this question positively. A few people thought that using news server will be useless for them and a few users learnt news server concept after our description and they found it as a useful system.

4. Internet bağlantınız böyle bir sisteme rahatlıkla ulaşacak kadar hızlı mı?

□ Evet. □ Hayır.

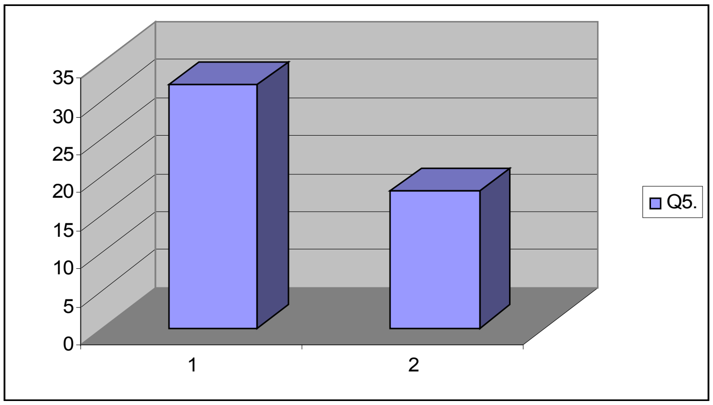


Most of the people have enough requirements for reaching a news server, so *NewsAgent* will be an applicable product for most of the people.

5. Böyle bir haber sunucusundaki haberlere ne şekilde ulaşmak istersiniz?

□ Sisteme giriş yaparak haberleri takip etmek

□ Haberleri e-mail yoluyla almak

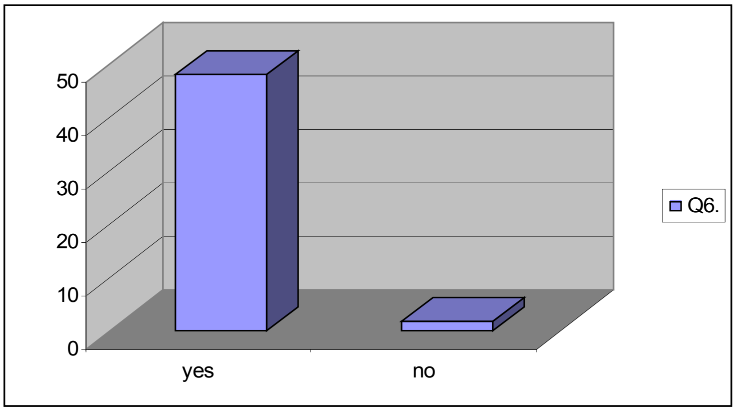


64% of people wanted to reach the news by logging in the system, since they want to prevent their mail-boxes from too many mails from newsgroups. The remaining 36% want to receive news via e-mail, because they check their mail-boxes periodically.

6. Haber sunucusundaki iletilerin, iletinin içeriğine göre farklı gruplar altında toplanmasını ister misiniz?

   □ Evet.

   □ Hayır, tüm iletileri içeren tek bir grup görmek isterim.

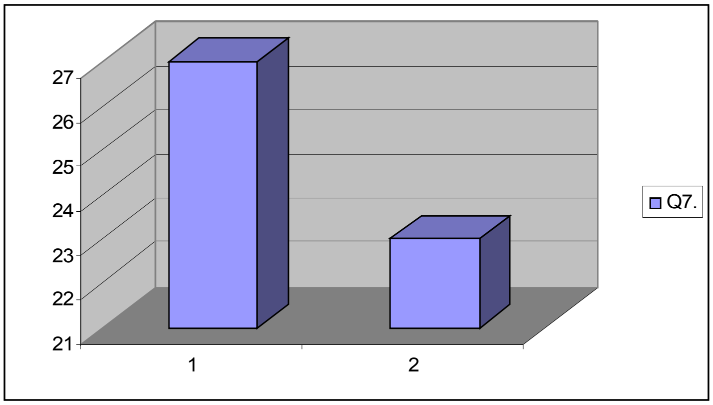


Almost all of the people want the articles to be divided into subgroups in order not to deal with articles that do not interest them. We also decided to divide the news into subgroups since it is more efficient and consistent.

7. İletilerin ekrandaki sıralamalarının neye göre yapılmasını istersiniz?

□ İletinin ilk atıldığı tarihe göre

□ İletiye ait son atılan cevabın tarihine göre

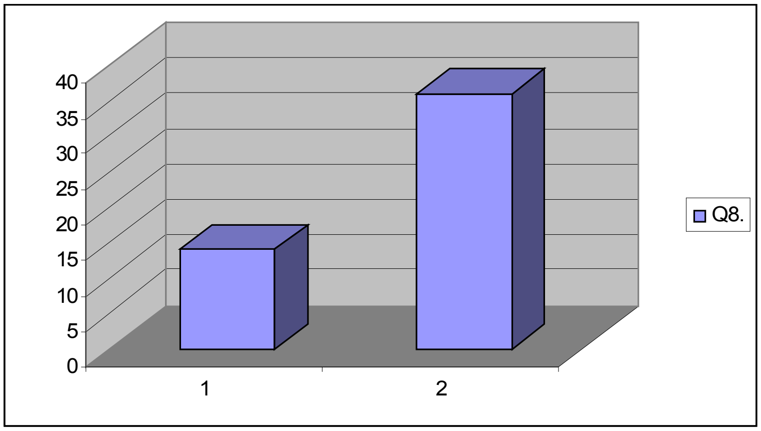


Since the percentage of people is almost equal for these two choices, we should provide these listing options together in our system.

8. Arşivleme işleminin belirli zaman aralıklarıyla otomatik olarak yapılmasını mı yoksa mesaj yüküne göre belirlenmesini mi istersiniz?

□ Otomatik olarak yapılmasını

□ Mesaj yüküne göre belirlenmesini



Since the message intensity differs from group to group, it is logical to archive articles according to message density. It balances the number of articles in subgroups. Users' opinions support our decision since most of them have chosen this option.

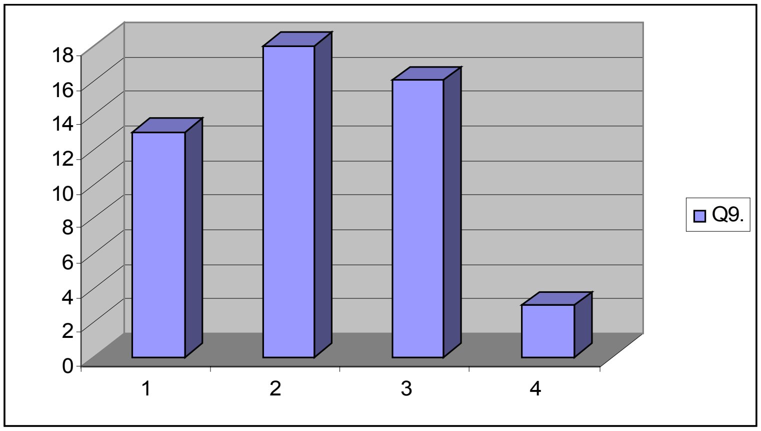
9. Üye olduğunuz haber gruplarının listelenmesinin neye göre yapılmasını istersiniz?

□ Alfabetik sıralama

□ Kullanıcıların grupları takip sırasına göre sıralama

□ Son gelen iletinin tarihine göre sıralama

□ Toplam ileti sayısına göre sıralama



Order of newsgroups can be seen insignificant at first glance, however, since our aim is to create a platform where users can reach data easily and in a consistent manner, user should be able to change the order of newsgroups as he wants. As shown in the statistics of this question, users want to have different kinds of order among newsgroups, consequently, it is reasonable to let user specify the order of newsgroups in the web component of *NewsAgent*.

10. Haber grubunu en çok nereden takip etmek istersiniz?

□ Web (www)

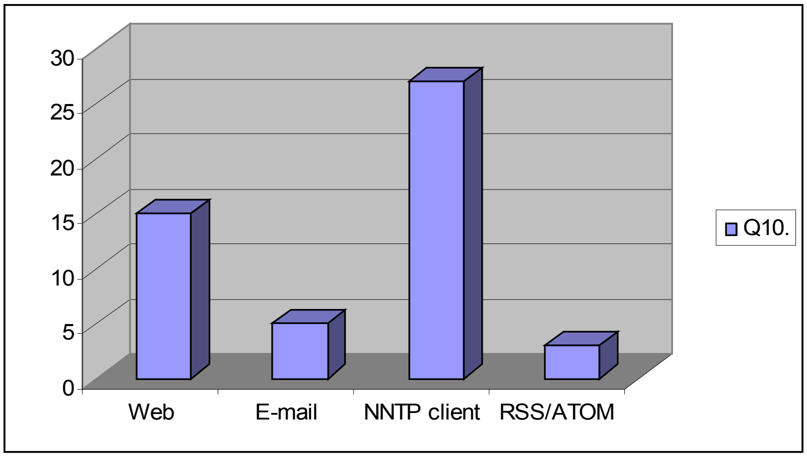
□ Outlook Express

□ Terminal (tin)

□ e-mail

□ RSS

□ Thunderbird



As it can be seen from the graph above, most of the possible users think it will be better to use an NNTP client for reaching newsgroups. Since we will implement all four methods specified on the graph, this question provides us to learn by which method, users will use our web service.

4 PROJECT REQUIREMENTS

4.1 SYSTEM REQUIREMENTS

4.1.1 SOFTWARE REQUIREMENTS

- Java as a programming language
- Eclipse as development environment
- Apache HTTP server
- Apache Tomcat for Servlet Container
- Apache Axis for XML XML Web Container
- PostgreSQL Database Management System
- Hibernate for Object-Relational Database Management. By this way, objects that we have created using Java can be inserted directly.

4.1.2 HARDWARE REQUIREMENTS

- For Developer
  A minimum of 512 MB DDRAM
  A minimum of 5 GB free space on hard disk, for database storage and server applications
  A Pentium IV processor
  Internet Connection
  Network Card

- For Server Applications
  A minimum of 1 GB DDRAM
  A minimum of 50 GB free space on hard disk, for huge database storage and large number of server applications
  A Pentium IV processor
  Internet Connection

4.2 FUNCTIONAL REQUIREMENTS

As a result of our market research, we have decided our functional requirements.

4.2.1 NewsAgent Core Requirements

Our main core deals with the article management. This core will mainly implement the USENET News Server. We have chosen the "pull" model that if any new article comes in, our server will only save the article and do the corresponding configurations. The clients can only be aware of the new message by checking the server if there is any new messages. Mail clients are out of this scope because our triggers in the database will automatically send the new articles to the mail clients subscribed to that news group with their mail addresses. Our articles will be stored in the database and in our system indexing of the articles, categorization of the articles into newsgroups, cross-referencing, archiving of old articles and also indexing of the archived articles will be provided.

Articles will be hold in a single table and there will be tables for all newsgroups holding pointers to those articles. Moreover, main core provides article and header based operations such as posting, reading, retrieving etc. For each of the article management operations, system will provide functionality which is implemented as web services. These web services will form the only access points to the main core. And they will be called from our modules. The related requests and commands will be mapped to the related web services for the operation functionalities. Below, the functional requirements for NewsAgent server core are listed. These functional requirements implement the NNTP Base Commands [4]. These commands can be found in APPENDIX A.

- ✓ Retrieve Article
- ✓ Retrieve Article Headers
- ✓ Retrieve Article Body
- ✓ Retrieve Article Statistics
- ✓ Select Newsgroup
- ✓ Get Help Information
- ✓ Offer Article To Server
- ✓ Go To Last Message
- ✓ List Newsgroups
- ✓ List New Newsgroups
- ✓ List New News Articles
- ✓ Go To Next Message
- ✓ Post Article
- ✓ End Session
- ✓ Set Slave Status

Web Service functionalities of NewsAgent are listed above. These services will also return some response to the clients that call them. These responses are the NNTP reply codes. These reply codes can be found in APPENDIX D.

The remaining part of the core handles the archiving and access level operations. Archiving functions will run due to some heuristic which considers the load of the articles in the corresponding tables.

4.2.2 Module Requirements

Web Module

In our web interface, we will display some newsgroups which can be accessed by authenticated and unauthenticated users. Unauthenticated users will only request to read the articles in these groups. If the user is unregistered, a sign up will be requested to get authenticated. If the user is registered, the following functionalities will be provided.

- The user logins to the system by entering username and password and after authentication check, the user group of the user is specified and the user will have the rights according to the user group. Each user group will have different rights and restrictions.
- Users can sign up only through web module and a randomly generated password is sent to the user via e-mail for verification of the candidate user. After the verification, user can start to reach articles on the news server by using his/her username & password.
- Update user info and account info functionalities will be supported and the user will be able to change this information.
- Read, post, update, cancel article functionalities will be provided and the user will have the right to update and cancel only the articles that he/she has posted.
- Mail receiving options will also be adjusted in the web module and a user may request to receive e-mail for the articles of the adjusted newsgroups.
- Listing the newsgroups and sorting the newsgroups and articles according to the specified criteria such as according to names, dates of the articles etc.

NNTP Module

Similar to the Web module, users are classified as authorized and unauthorized users. Unauthorized users can only reach only some subset of newsgroups, which are specified by system administrator. In fact, that is reasonable, since user group of unauthorized users has access level to only these newsgroups. If user is registered, the following functionalities will be provided to the user:

- The user login to the system by entering his/her username and password. Username and password are controlled for validation from the database. If username-password combination is not valid, display feedback is shown to the user which specifies incorrect username or password and user cannot enter the system. If this is not the case, user can enter the system and an access level is assigned to the user corresponding to the user group.
- User can update his/her account information according to his wishes. Since userid information is hidden from the user, same userid will again specify the user.
- Read, post, update, cancel article functionalities will be provided and the user will have the right to update and cancel only the articles that he/she has posted.
- Mail receiving options will be adjusted in the NNTP Module, setting this option on/off is the users' choice.
- Listing the newsgroups and sorting the newsgroups and articles according to the specified criteria such as according to names, dates of the articles etc.

Mail Module

- When our system receives an e-mail, first of all the system controls whether the sender is an authenticated mail client or not. If the sender is authenticated then the e-mail is converted to the article format and inserted to the database. The article will be added to a newsgroup which is specified in the address field of the mail content.
- User can reach articles in a newsgroup via e-mail depending on whether he/she set his mailing options on. Of course, user will be able to receive mail from only newsgroups which he/she can subscribe corresponding to his/her user group.

RSS/Atom Module

- If user want to follow a newsgroup periodically, user can subscribe to the RSS feed of this newsgroup and by using an RSS reader, he/she can reach articles that are newly posted to the newsgroup.

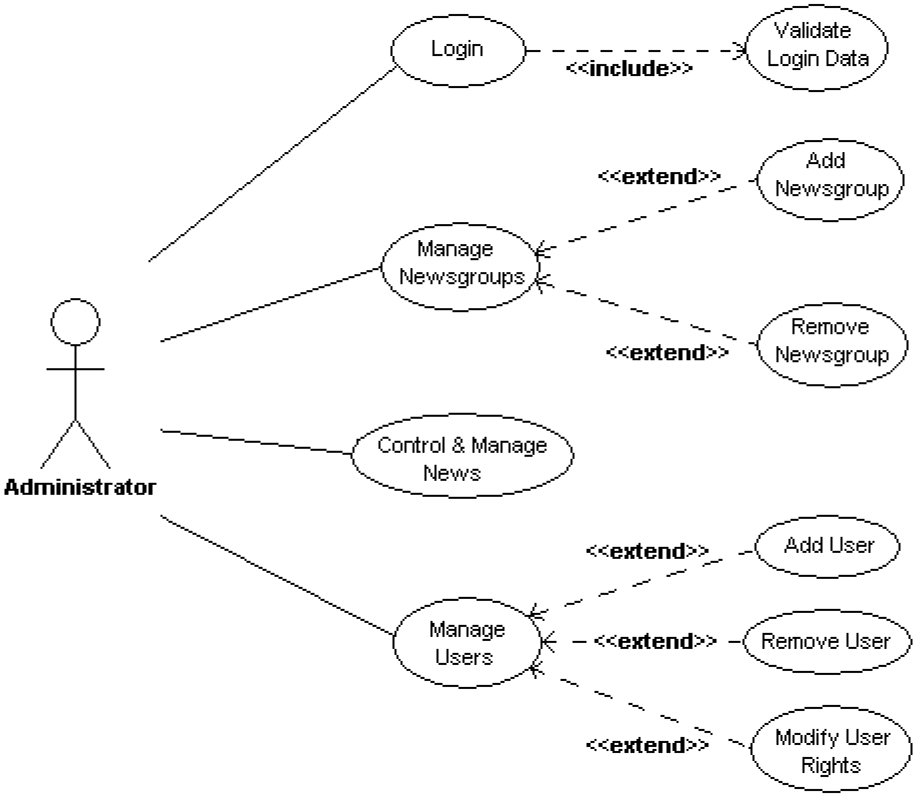
Authentication Module

- As mentioned in previous modules, each user will be in a user-group which specifies the access level of the user. During authentication username will be checked for specifying whether username is in database or not.
- Username and password will be checked for correspondence between them.
- For security reasons, password will be held in a MD5 (Message-Digest algorithm 5) [5] format. This hashing technique will prevent anyone to access passwords of the users, directly.
- User groups will be assigned for the user after his/her authentication. Since user group for each user is stored in the database which is assigned by system administrator, assignment of user groups is not a big deal.
- A user who is not authorized to the system will be able to access only some subset of newsgroups and read only articles in these newsgroups.

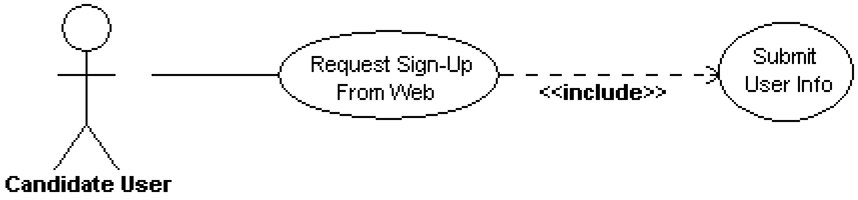
4.3 USER REQUIREMENTS

4.3.1 Use Case Diagrams

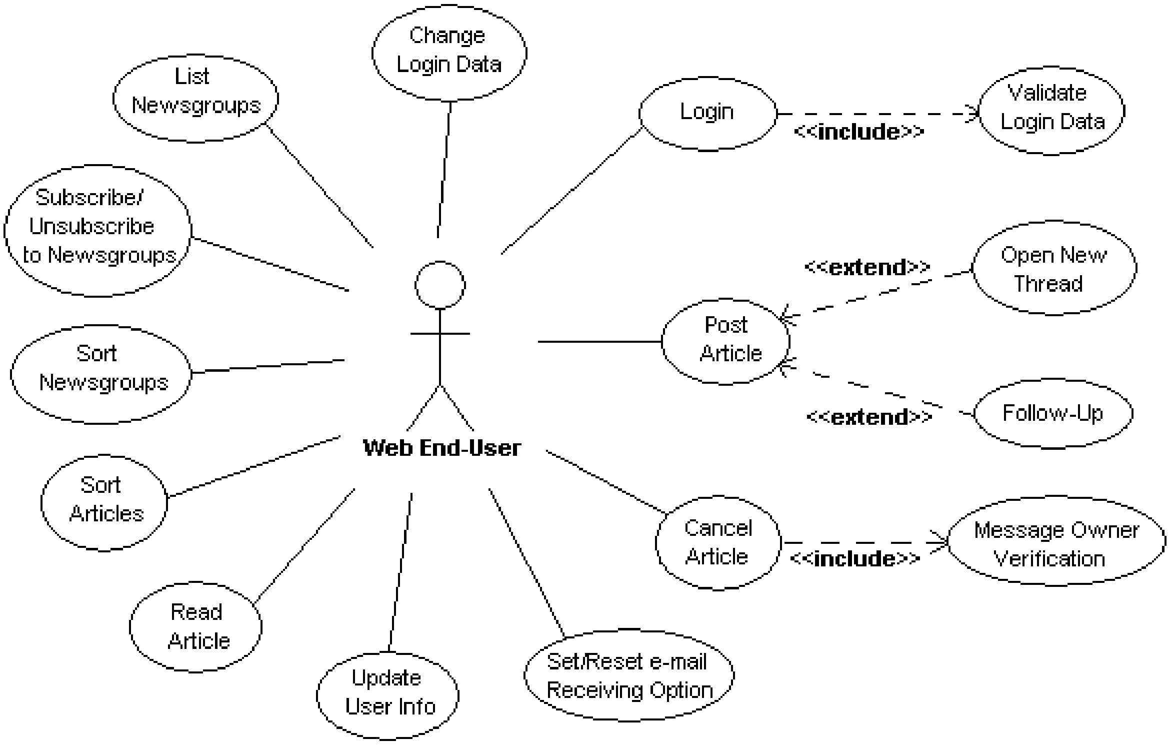
4.3.1.1 Use Case Diagram for Administrator



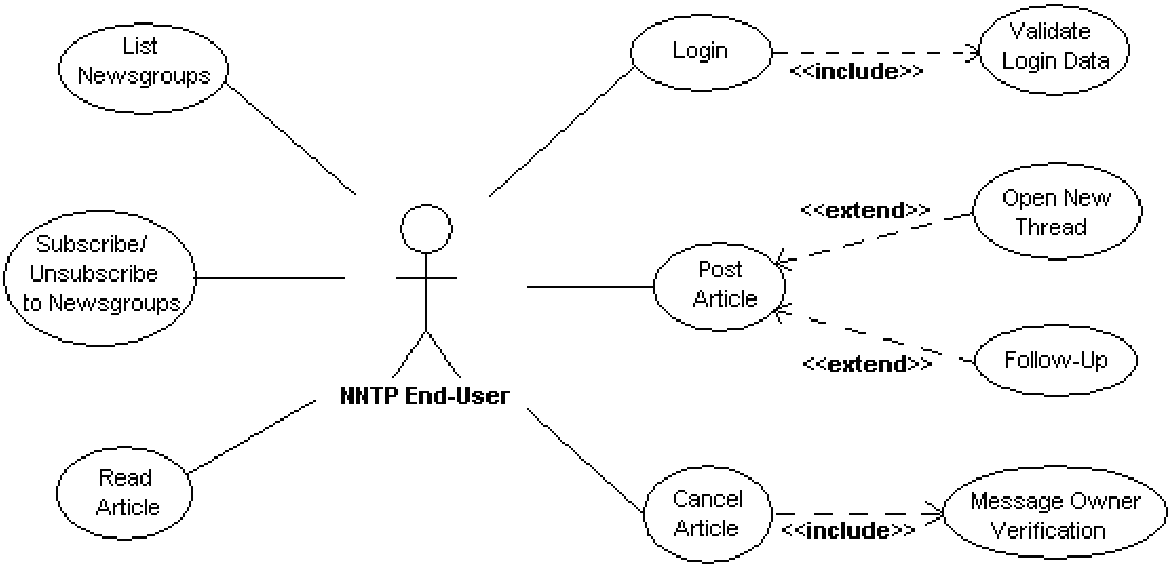
4.3.1.2 Use Case Diagram for Candidate User



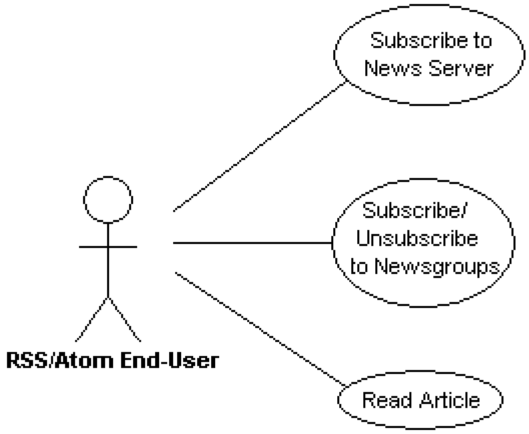
4.3.1.3 Use Case Diagram for Web End-User



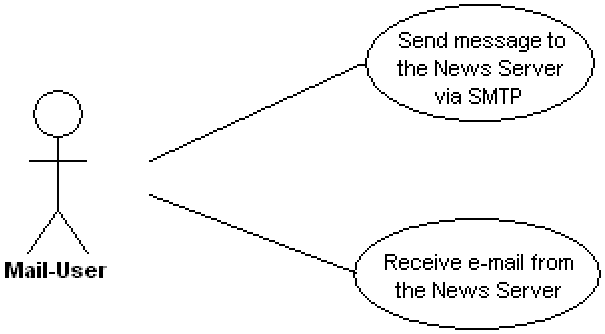
4.3.1.4 Use Case Diagram for NNTP End-User



4.3.1.5 Use Case Diagram for RSS/Atom End-User



4.3.1.6 Use Case Diagram for Mail-User


Send message to the News Server via SMTP
Receive e-mail from the News Server
Mail-User

4.3.2 Use Case Scenarios

<u>**Administrator:**</u>

- ***Login:*** An administrator has to login to the system in order to realize administrative roles. There will be a web user interface for administrative roles. After validation of login information, the administrator will be able to manage newsgroups, users and news.
- ***Manage Newsgroups:*** Administrator may add new newsgroups and remove existing newsgroups in the content of the managing newsgroups scenario.
- ***Manage Users:*** Administrator may add and remove users and modify the user rights. Administrator will control users and will be able to restrict the user rights.

There will be specified user roles and rights, however, new rights can be granted to the users and existing rights may be withdrawn.

- ***Control & Manage News:*** An administrator will have the right of controlling and managing the articles. Articles which do not suit the content of the newsgroup may be cancelled. As a result of such a control on news, user roles and rights granted to the users defined more precisely.

Candidate User:

- ***Request Sign-up:*** A candidate user is a person who demands to sign up to the system via web interface and as a result of a sign-up request, the candidate user has to submit a user information form and if the administrators accept the request, the candidate user turns out to be a real system user.

Web End-User:

The scenarios which are valid for NNTP End-users are also valid for Web End-users. Moreover, Web End-users have extra usage scenarios. The followings are the extra usage scenarios for Web End-users.

- ***Set & Reset Mail Receiving Options:*** The user will be able to request to receive e-mail for the articles posted. The user may want to receive e-mail for specified newsgroups or want to receive e-mail for all newsgroups. Also the user may want to cancel the mail receiving option and then no e-mails will be sent to the user from that newsgroup.
- ***Update User Info:*** The user will be able to update user information such as his/her personal information registered when signing up, e-mail address etc.
- ***Change Login Data:*** The user may change login information. Generally user id of a user is not allowed to be changed for most of the systems however the users may need to change their passwords.

NNTP End-User:

- ***Login:*** The user will login to the system in order to realize user roles. After validation of user login information, the user will be able to list, subscribe/unsubscribe, sort newsgroups and post, read, cancel and sort articles.
- ***List Newsgroups:*** The user will be able to list the newsgroups. In the concept of listing newsgroups scenario, a user may list all newsgroups or the newsgroups that he/she has been subscribed.
- ***Subscribe / Unsubscribe to Newsgroups:*** After listing the newsgroups, the user will be able to subscribe and unsubscribe to the newsgroups.
- ***Post Article:*** The user posts articles. In the concept of posting articles, the user may open a new thread or follow up to an existing article.
- ***Cancel Article:*** The user may cancel the articles that he/she has posted.
- ***Read Article:*** The user reads articles.

RSS/Atom End-User:

- ***Subscribe to News Server:*** RSS/Atom end-users will subscribe to the news server in order to receive feeds from the server.
- ***Subscribe / Unsubscribe to Newsgroups:*** RSS/Atom end-users will be able to subscribe and unsubscribe to specific newsgroups. Each newsgroup will have its own feed so that the user receives only the news from subscribed newsgroups.
- ***Read Articles:*** As all users do, RSS users will read the news.

Mail User

When a user sets receiving mail option from web, that user becomes also a mail user.

- ***Send Message to the News Server:*** Mail users send messages to the server through SMTP protocol. This message appears in the same way as other messages do in the News Server.
- ***Receive e-mail from the News Server:*** When a new message is posted, mail users receive that message as e-mail from the newsgroups if they are subscribed to that group.

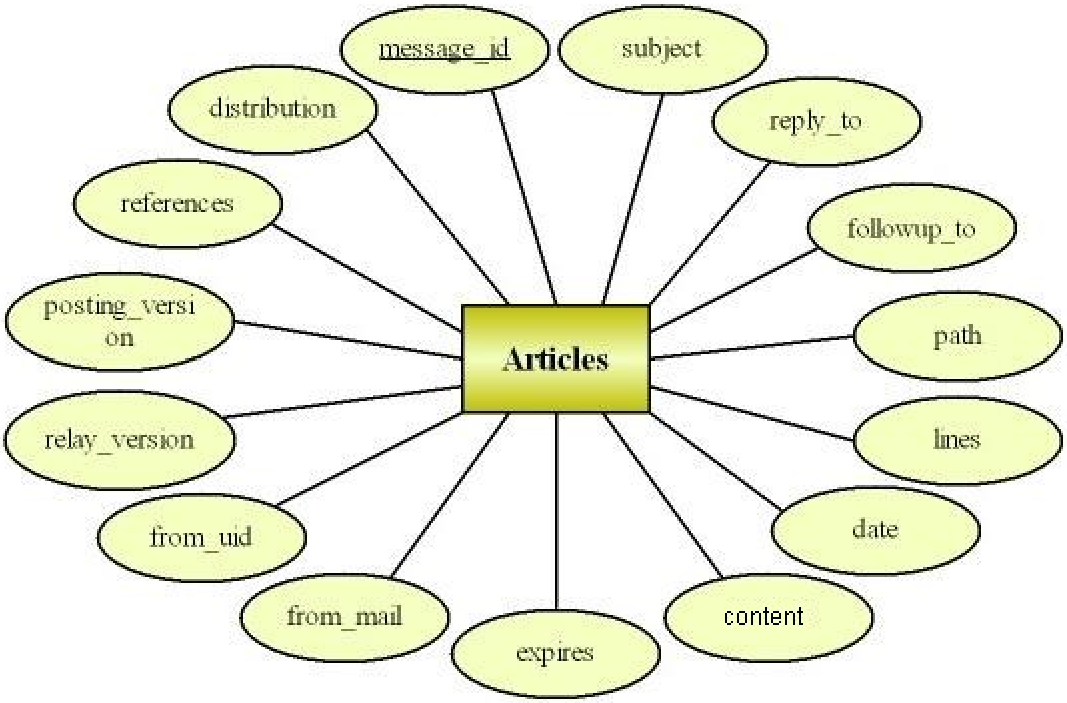
5 MODELLING

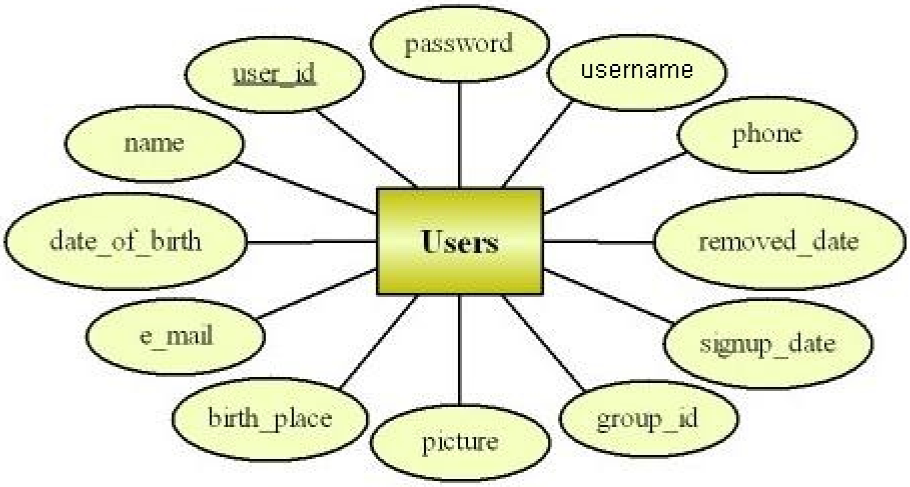
5.1 DATA MODELLING

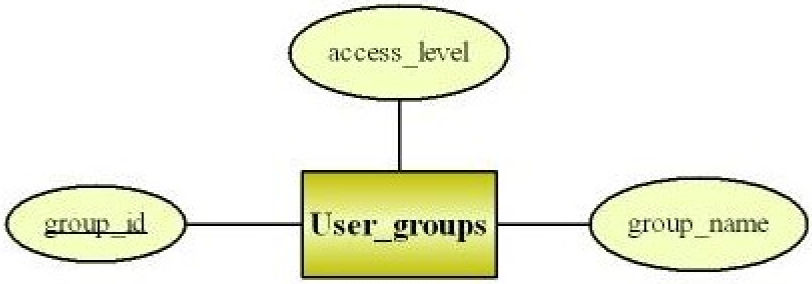
In our system, we will store our data in 2 different databases. The main database will be used to store main data such as articles, users, newsgroups, etc. Other database will be used as an archive to store older articles. These older articles will not be stored in main database anymore.

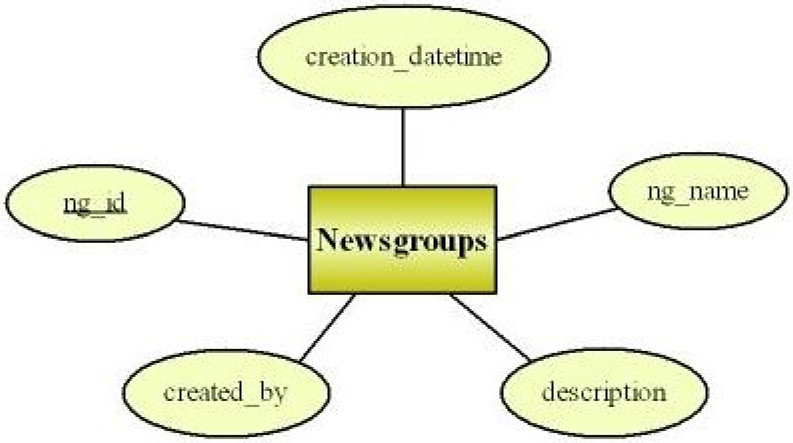
5.1.1 Entity-Relationship Diagrams

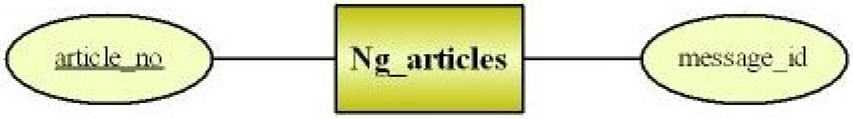
ER Diagrams for Main Database

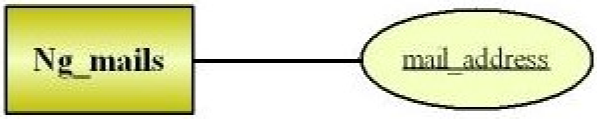


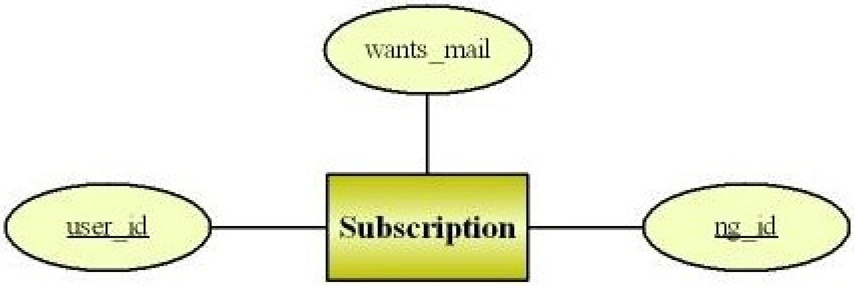

Users
password
user_id
username
name
phone
date_of_birth
removed_date
e_mail
signup_date
birth_place
picture
group_id


access_level
group_id
User_groups
group_name

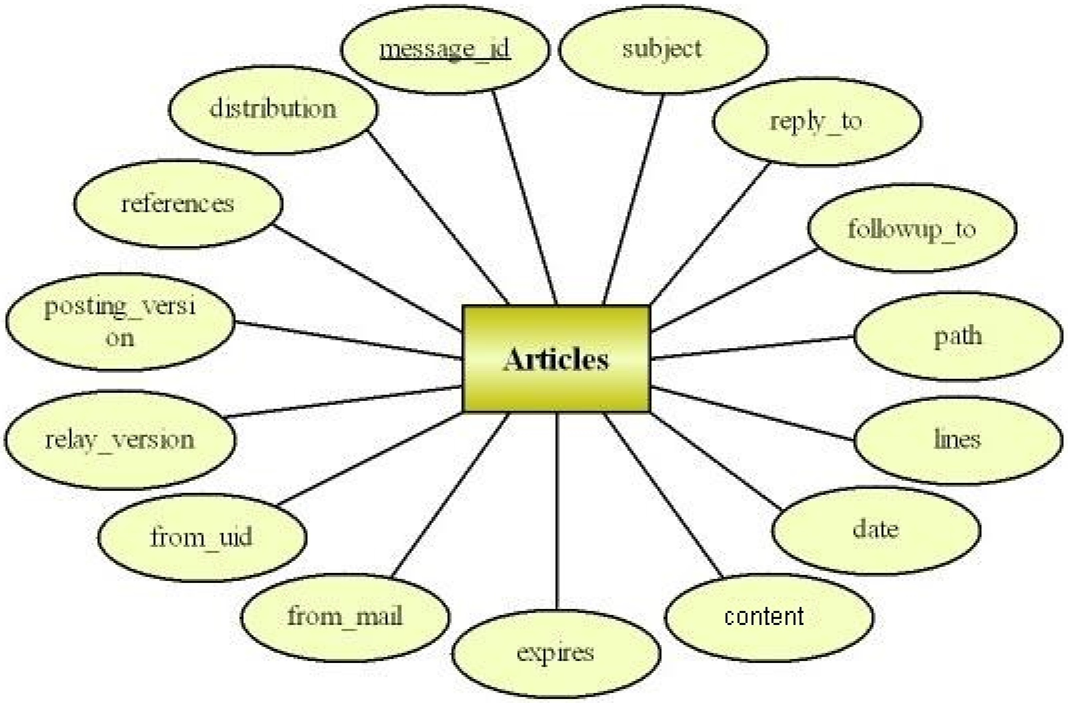

creation_datetime
ng_id
Newsgroups
ng_name
created_by
description

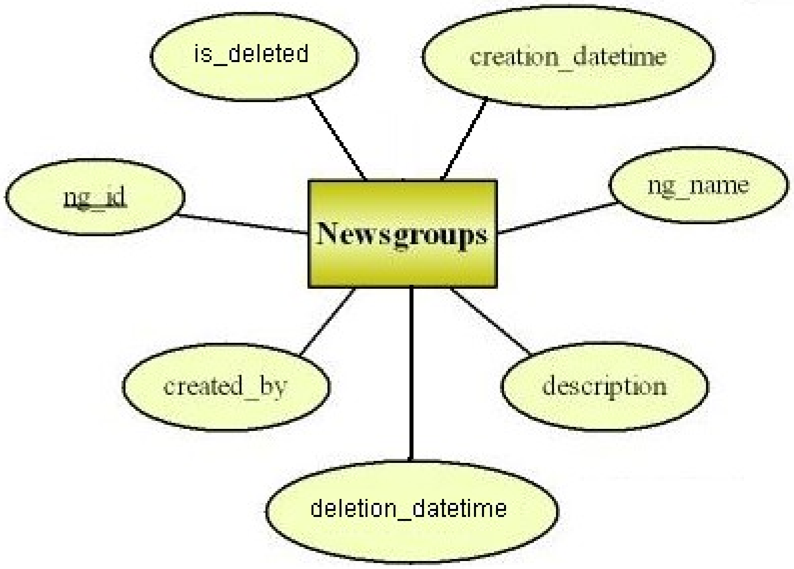

article_no
Ng_articles
message_id

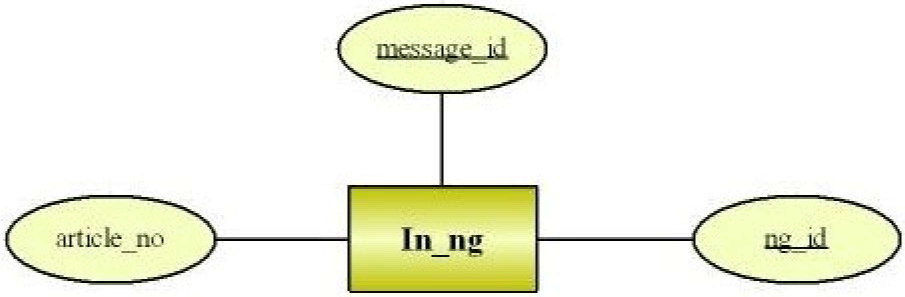

Ng_mails
mail_address


wants_mail
user_id
Subscription
ng_id

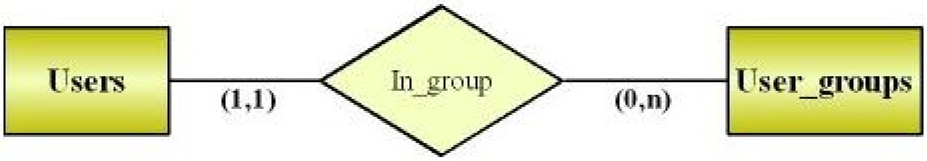
ER Diagrams for Archive Database

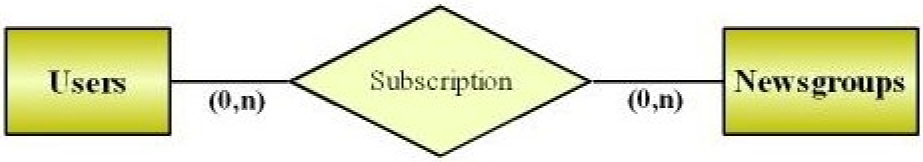


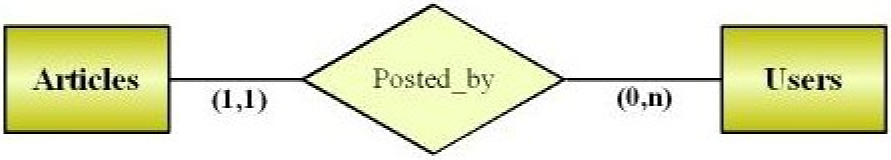


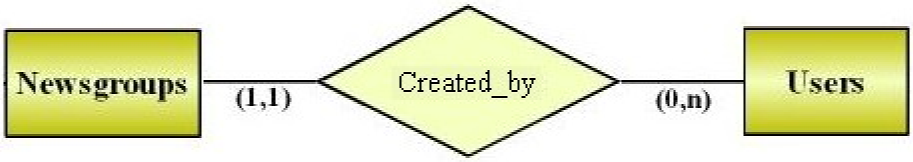

message_id
article_no
In_ng
ng_id

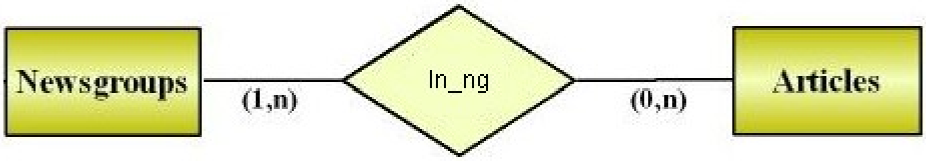
Relations


Users
(1,1)
In_group
(0,n)
User_groups


Users
(0,n)
Subscription
(0,n)
Newsgroups


Articles
(1,1)
Posted_by
(0,n)
Users


Newsgroups
(1,1)
Created_by
(0,n)
Users


Newsgroups
(1,n)
In_ng
(0,n)
Articles

5.1.2 Data Descriptions

The data description function is to deal with the structure of the data. We have taken each entity and relation separately and given each attribute in each entity or relation a type so the data is fully structured.

Note:

- **Data with underlines are primary keys;**
- **Data with star have to be entered absolutely (NOT NULL);**

Data Descriptions for Main Database

Articles

Data	Type & Size	Format
message_id*	VARCHAR – 40	Text
subject*	VARCHAR – 60	Text
content	TEXT	Text
date*	DATETIME	Date/time
from_uid*	BIGINT	Number
from_mail*	VARCHAR – 40	Text
reply_to	VARCHAR – 40	Text
followup_to	VARCHAR – 40	Text
relay_version*	VARCHAR – 60	Text
posting_version*	VARCHAR – 60	Text
lines*	INTEGER	Number
path*	VARCHAR – 60	Text
expires	DATETIME	Date/time
references	VARCHAR – 60	Text
distribution	VARCHAR – 60	Text
control	VARCHAR – 60	Text

Users

Data	Type & Size	Format
user_id*	BIGINT	Number
password*	VARCHAR – 20	Text is hidden. ********
name*	VARCHAR – 40	Text
username*	VARCHAR – 40	Text
date_of_birth	DATE	Date
birth_place	VARCHAR – 20	Text

phone*	VARCHAR – 40	Text
e-mail*	VARCHAR – 40	Text
signup_date*	DATETIME	Date/time
removed_date	DATETIME	Date/time
group_id*	INTEGER	Number
picture	BLOB	Binary

User_groups

Data	Type & Size	Format
group_id*	INTEGER	Number
group_name*	VARCHAR – 60	Text
access_level*	INTEGER	Number

Newsgroups

Data	Type & Size	Format
ng_id*	INTEGER	Number
ng_name*	VARCHAR – 60	Text
created_by*	BIGINT	Number
creation_datetime*	DATETIME	Date/time
description	VARCHAR – 60	Text

Ng_articles

Data	Type & Size	Format
article_no*	BIGINT	Number
message_id*	VARCHAR – 40	Text

Ng_mails

Data	Type & Size	Format
mail_address*	VARCHAR – 40	Text

Subscription

Data	Type & Size	Format
user_id*	BIGINT	Number
ng_id*	INTEGER	Number
wants_mail*	BOOL	Yes/no

Data Descriptions for Archive Database

Articles

Data	Type & Size	Format
message_id*	VARCHAR – 40	Text
subject*	VARCHAR – 60	Text
content	TEXT	Text
date*	DATETIME	Date/time
from_uid*	BIGINT	Number
from_mail*	VARCHAR – 40	Text
reply_to	VARCHAR – 40	Text
followup_to	VARCHAR – 40	Text
relay_version*	VARCHAR – 60	Text
posting_version*	VARCHAR – 60	Text
lines*	INTEGER	Number
path*	VARCHAR – 60	Text
expires	DATETIME	Date/time
references	VARCHAR – 60	Text
distribution	VARCHAR – 60	Text

Newsgroups

Data	Type & Size	Format
ng_id*	INTEGER	Number
ng_name*	VARCHAR – 60	Text
created_by*	BIGINT	Number
is_deleted*	BOOLEAN	Yes/no
creation_datetime*	DATETIME	Date/time
deletion_datetime	DATETIME	Date/time
description	VARCHAR – 60	Text

In_ng

Data	Type & Size	Format
message_id*	VARCHAR – 40	Text
ng_id*	INTEGER	Number
article_no*	BIGINT	Number

5.1.3 Entity Descriptions

Entity & Relation Descriptions for Main Database

Articles

This entity contains all necessary information about articles which are posted to the news server. No matter to which group it is posted, all articles are stored in this table with all required information. Some attributes are used for holding standard data for USENET messages and some attributes are assigned by us locally for managing articles easily.

In USENET message format, [6] there are some required headers and some optional headers. We hold these required headers and some of the optional headers in our database, in *Articles* entity, in order to obey universal USENET message standards. Below, the table's attributes are explained.

***message_id**:** Required \`Message-ID\` standard header is held in string *message_id*. This attribute is the primary key of *Articles* entity and uniquely defines a message. The same message ID may not be assigned to another article during the lifetime of that article.

***subject**:** Required \`Subject\` standard header is held in string *subject*. It is assigned by sender and briefly defines what the article is about.

content: This field is held in text format and stores the content of the article.

***date**:** Required \`Date\` standard header is held in *date* in date/time format. It is the time that the article is posted to the network.

***from_uid**:** This is a local assignment that is required to know which user has posted the article. It is a foreign key for this entity referencing *user_id* of *Users* entity.

***from_mail**:** Required \`From\` standard header is held in string *from_mail.* It is the mail address of the sender of that article. This is a default mail address and foreign key which references the attribute *e-mail* of *Users* entity.

reply_to: Optional \`Reply-To\` standard header is held in string *reply_to*. This string holds the optional mail address of the sender if he/she wants to get mail for that article to the specified address instead of *from_mail*.

followup_to: Optional \`Followup-To\` standard header is held in string *followup_to.* If this is not empty, all follow-ups to the article will be posted to the newsgroups specified in this field. If it is empty, follow-ups will be posted to the newsgroup(s) that the message was originally posted.

relay_version*: Required `Relay-Version` standard header is held in string *relay_version*. This header shows the version of the program that is responsible for the transmission of the article.

posting_version*: Required `Posting-Version` standard header is held in string *posting_version*. This header identifies the software that is responsible for passing this message into the network.

lines*: This header is also required and specifies how many lines the article has. It is held in integer format.

path*: Path is a required header and shows the way that the article followed until reaching the system. Path is held in string format and when a system forwards this article, it concatenates its name to the path.

expires: This field is in date/time format and optional. If it exists, the article expires in specified date and time.

references: This field is optional and held in string format consisting of article ID`s which prompt the submission of this article. For instance, in a follow-up article, the parent article exists in this field.

distribution: This field is held in string format and lists the newsgroups that the article should be sent. This field alters the original newsgroup distribution.

Users

This entity contains all required information about the users which can be authorized or unauthorized. Administrators are also users.

user_id*: This number specifies each user uniquely; hence *user_id* is the primary key of the *Users* entity.

name*: This string field holds the name of the user.

username*: This string field holds the username of the user.

password*: This string field is the matched password for the username of the user .

date_of_birth: This date typed attribute holds the birth date of the user.

birth_place: This string typed attribute holds the birth place of the user.

phone*: This string field holds the cell phone number of the customer.

e-mail*: This text field holds the mail address of the customer.

signup_date*: This field holds the date and time that the user has signed up. This field is of type date/time.

removed_date: This field is usually empty but if a user is removed from the database, this field holds the date and time that the user is removed from the system.

group_id*: Group id specifies which user group the user belongs to. This is a foreign key referencing *group_id* attribute of *User_groups* entity.

picture: Users can upload their pictures to the system. This picture is held in *picture* field in BLOB format.

User_groups

This entity holds information about user groups. Users are assigned to user groups according to their access rights.

group_id*: This number specifies each user group uniquely; hence *group_id* is the primary key of the *User_groups* entity.

group_name*: This string field holds the name of the usergroup.

access_level*: This integer field holds the access level of the user. For instance, if it is 1, it means full access.

Newsgroups

This entity holds information about newsgroups. When a newsgroup is added, listed information about that group is added to the table.

ng_id*: This number specifies each newsgroup uniquely; hence n*g_id* is the primary key of the *Newsgroups* entity.

ng_name*: This string field holds the name of the newsgroup.

created_by*: This big integer typed field holds information about who created this newsgroup. This is a foreign key of this entity referencing *user_id* attribute of *Users* entity.

creation_datetime*: This field holds the date and time that the newsgroup is created. This field is of type date/time.

description: This string field holds a brief description about what the newsgroup is about.

Ng_articles

Ng_articles is a general name for lots of possible tables. When a new newsgroup is created, an article table is created for that newsgroup with a specifying name. For example, if a group named `Music` is created, a table named `Music_articles` is also created. This table does not hold all information about the articles belonging to that table. It only holds little information about articles posted to that group for referencing the articles from main *Articles* table. This way is chosen in order to prevent the database from multiple storage of same article when it is posted to different groups at the same time.

article_no*: This number specifies each article in that newsgroup uniquely; hence article_no is the primary key of the *Ng_articles* entity.

message_id*: This is a foreign key referencing *message_id* attribute of *Articles* entity. Original messages are referenced by this field.

Ng_mails

Ng_mails is also a general name for lots of possible tables. When a new newsgroup is created, a mails table is created for that newsgroup with a specifying name. For example, if a group named `Cinema` is created, a table named `Cinema_mails` is also created. This entity is formed in order to store mail addresses of people who subscribed to receive the articles that are posted to the specified newsgroup as e-mail.

mail_address*: This string field holds the mail addresses of the users who want to receive e-mails from the specified newsgroup.

Subscription

This table specifies a relation among users and newsgroups. Users can be subscribed to newsgroups. Required information about this subscription is held in this table.

user_id*: This field is the id of the user who subscribed to the newsgroup. This is a foreign key for this relation referencing *user_id* attribute of *Users* entity. This field is a subset of primary key.

ng_id*: This field is the id of the newsgroup which is subscribed by the user. This is a foreign key for this relation referencing *ng_id* attribute of *Newsgroups* entity. This field is also a subset of primary key.

- ng_id and user_id are primary key of the relation together.

wants_mail*: This Boolean type is hold to know whether the user wants e-mail from this newsgroup or not.

Entity Descriptions for Archive Database

Articles

This entity is the same as *Articles* entity in main database. Definitions of attributes are as listed there.

Newsgroups

This entity is the same as *Articles* entity in main database except for the *is_deleted and deletion_datetime* attributes of this newsgroups entity. *is_deleted* boolean attribute specifies whether that newsgroup is deleted or not, since a deleted newsgroup can exist in archive database but not main database. *deletion_datetime* attribute specifies the deletion time of the newsgroup if it is deleted. Definitions of other attributes are as listed in definition of main database entity.

In_ng

This table specifies a relation among articles and newsgroups in archive database. Articles belong to newsgroups. We needed this relation only for this database, since in archive database; we do not hold different tables for different newsgroups that list the articles posted to that newsgroup.

message_id*: This field is a foreign key for this relation referencing *message_id* of *Articles* entity. It defines which message is in the newsgroup.

ng_id*: This field is a foreign key for this relation referencing *ng_id* of *Newsgroups* entity. It defines which newsgroup the message belongs to.

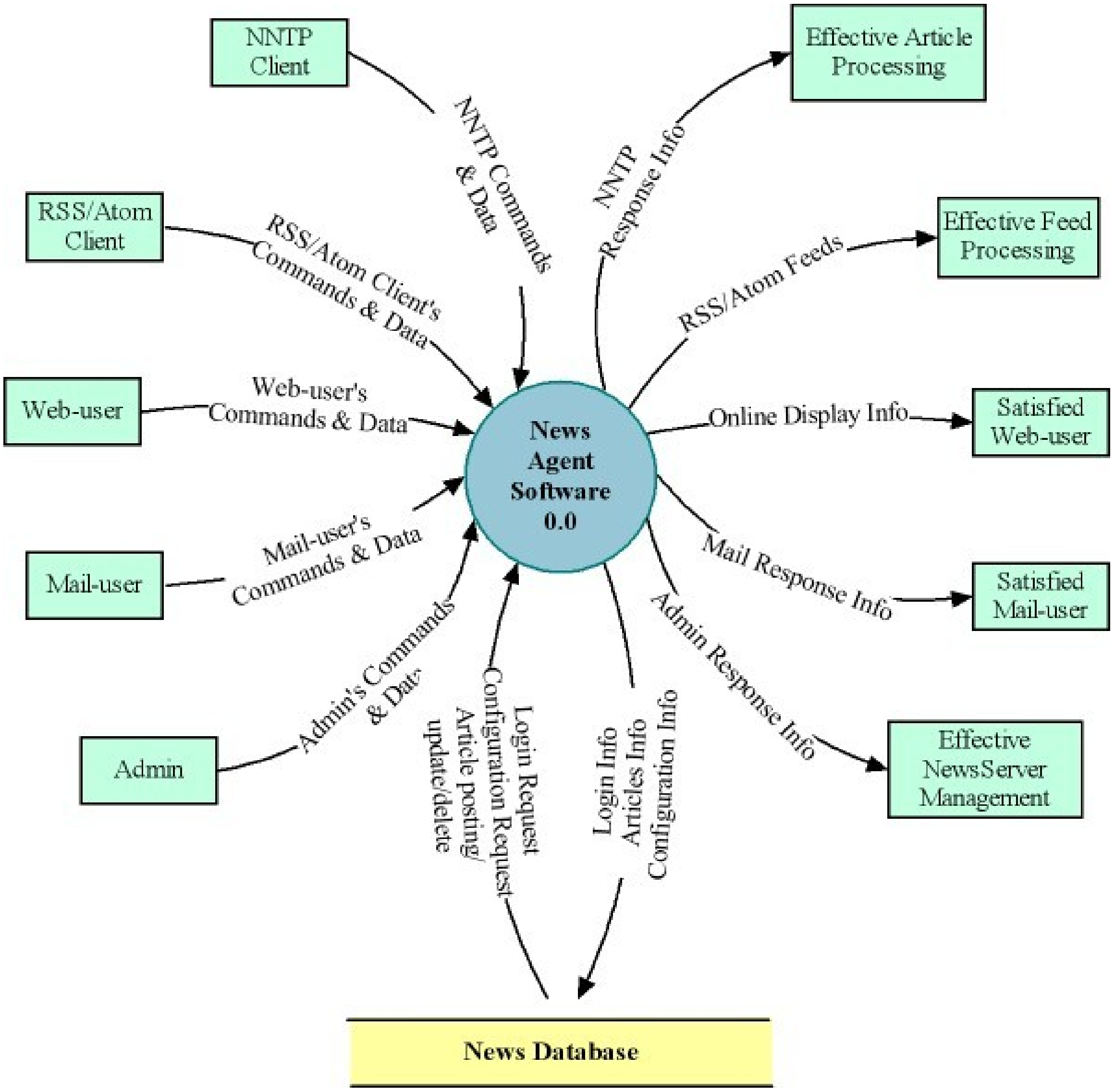
- ng_id and message_id are primary key of the relation together.

article_no*: This number is assigned to the articles that take place in this relation.

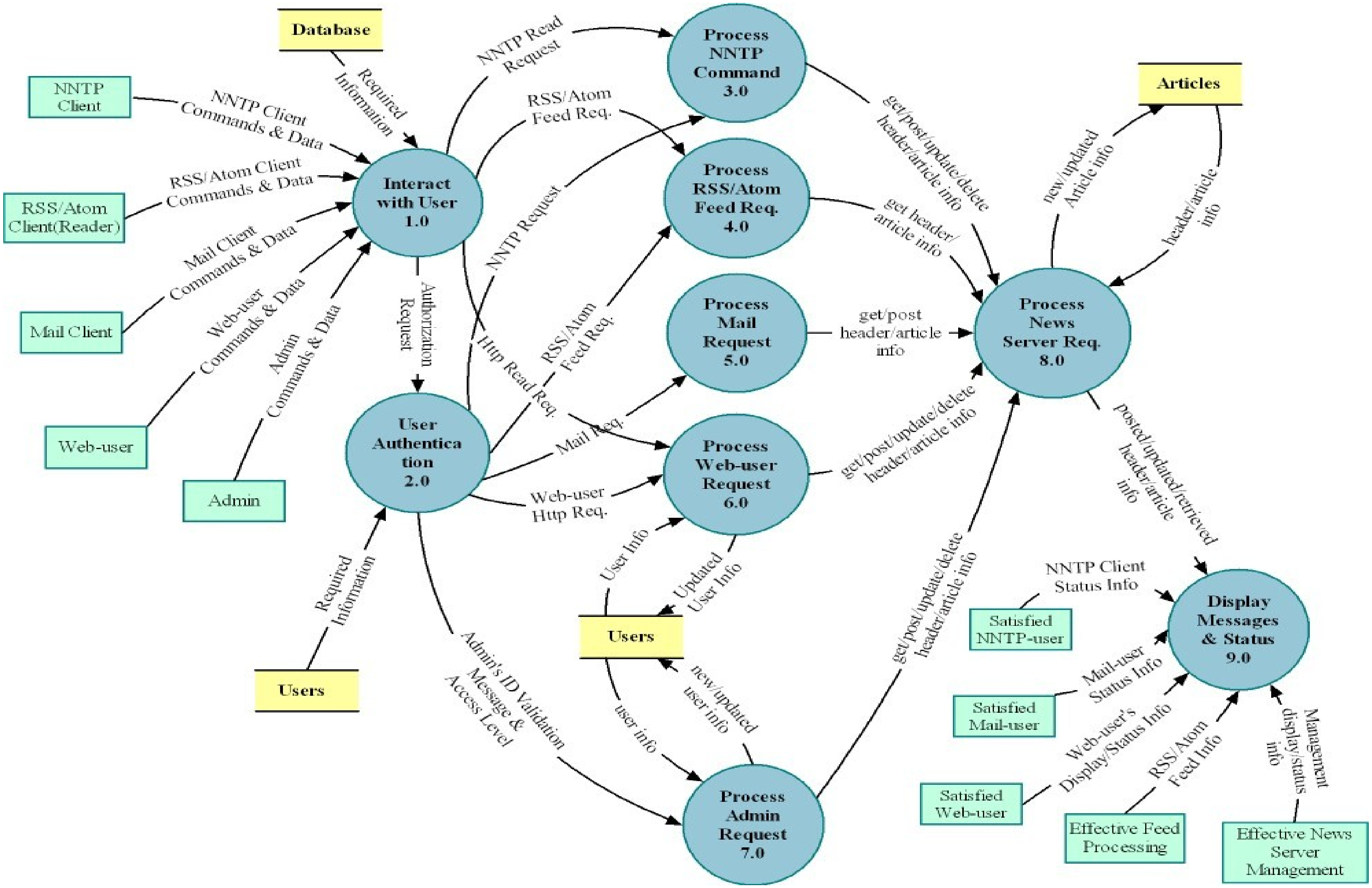
5.2 FUNCTIONAL MODELLING

5.2.1 Data Flow Diagrams

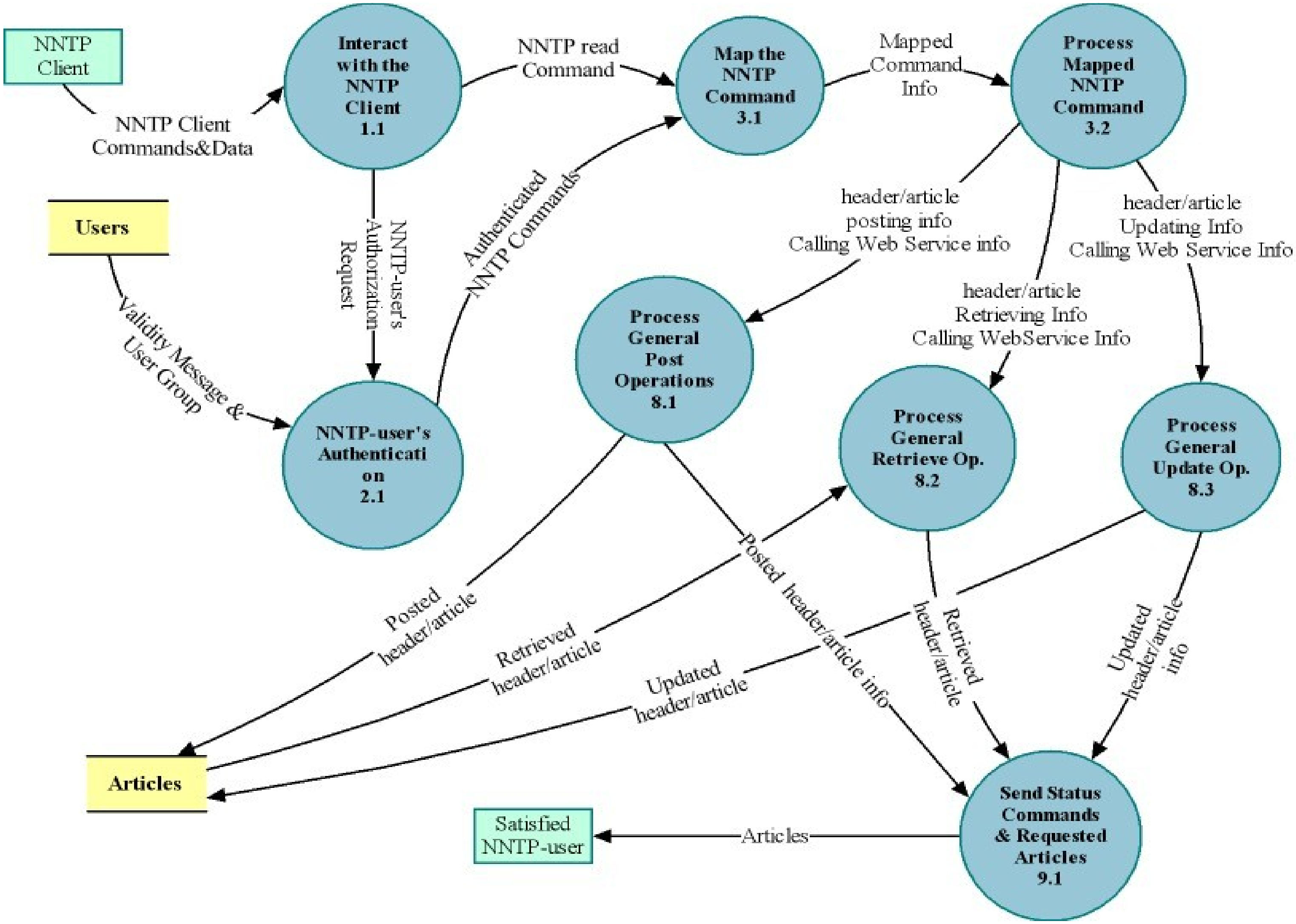
5.2.1.1 Level 0 Data Flow Diagram

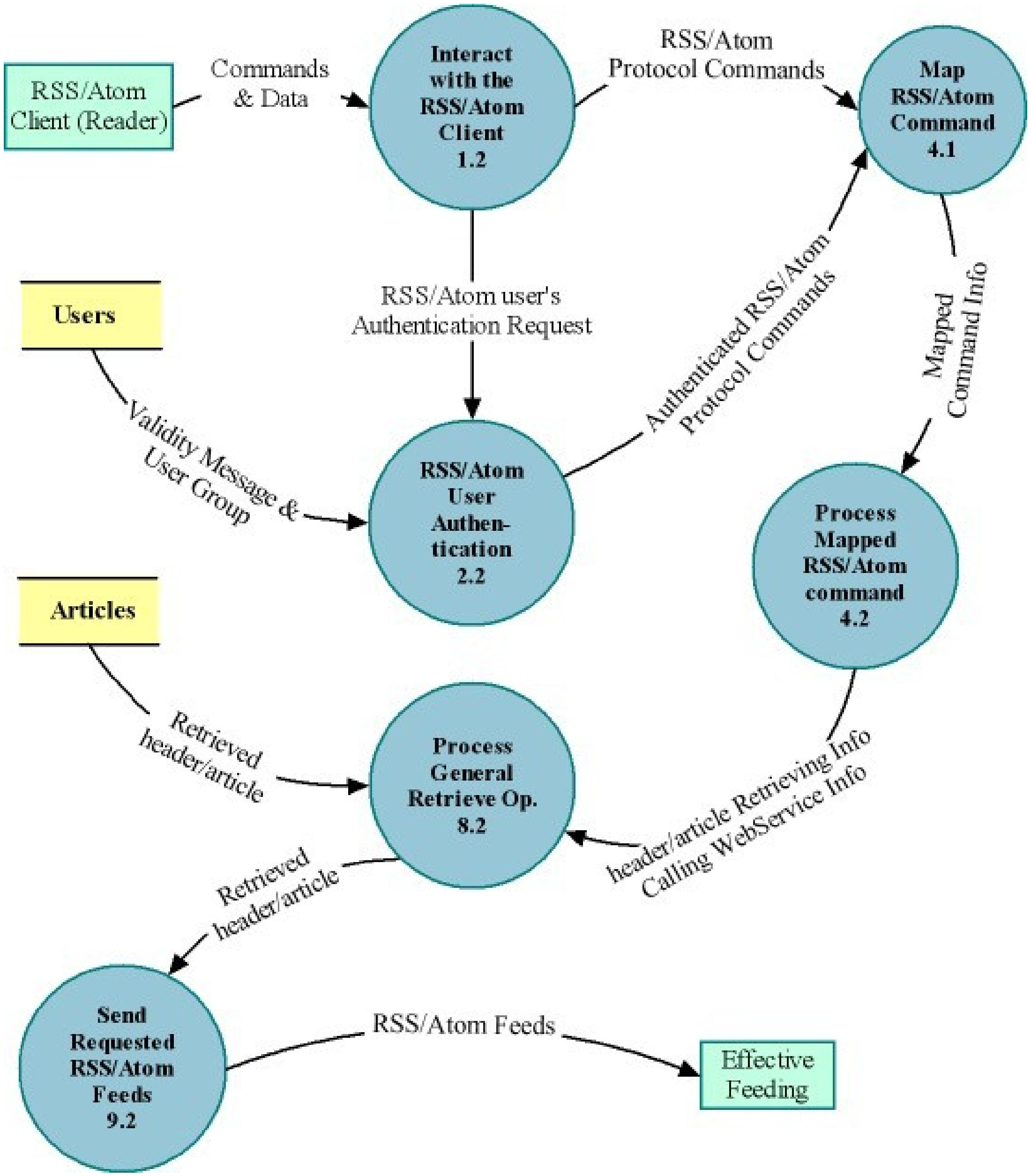


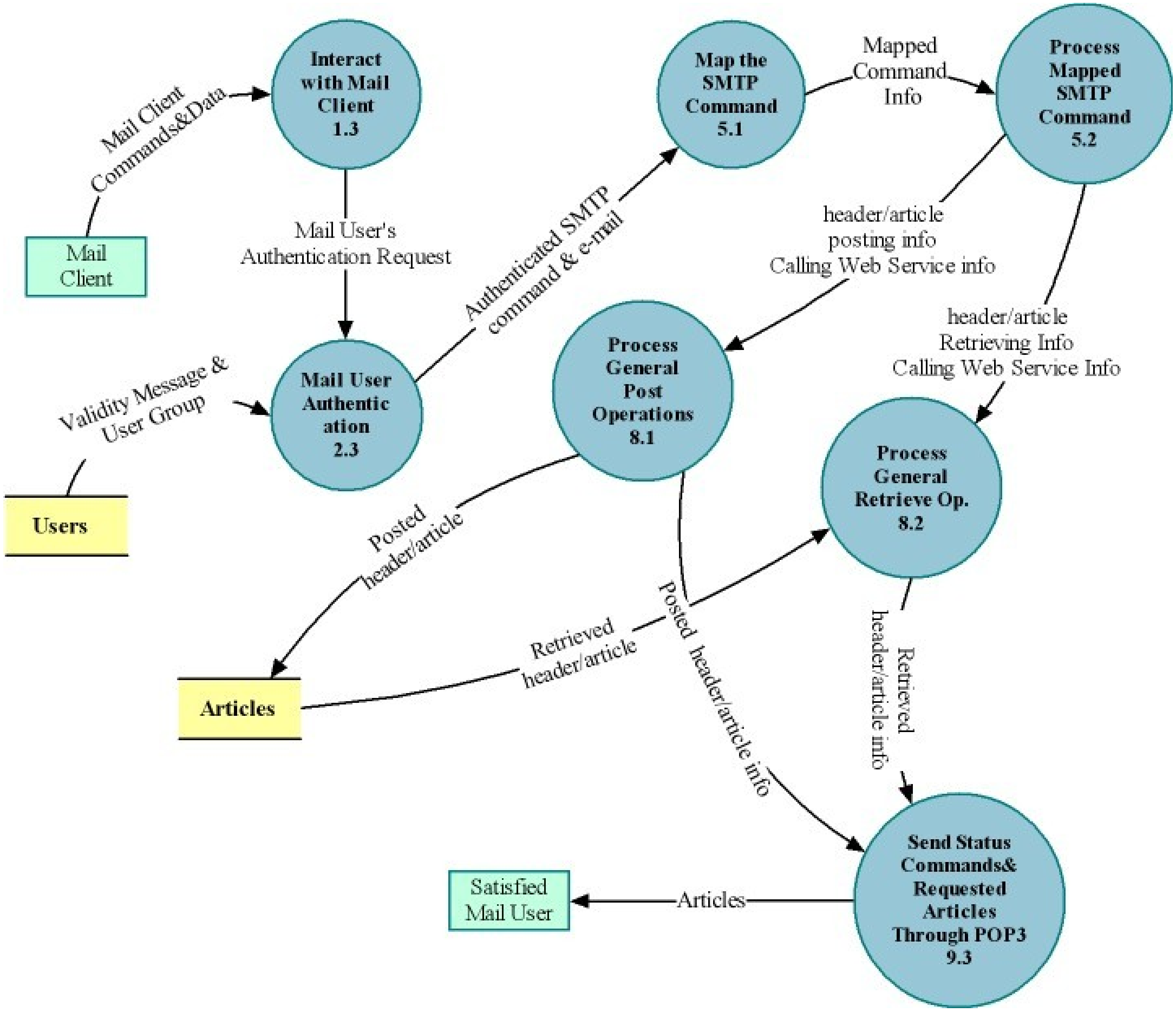
5.2.1.2 Level 1 Data Flow Diagram

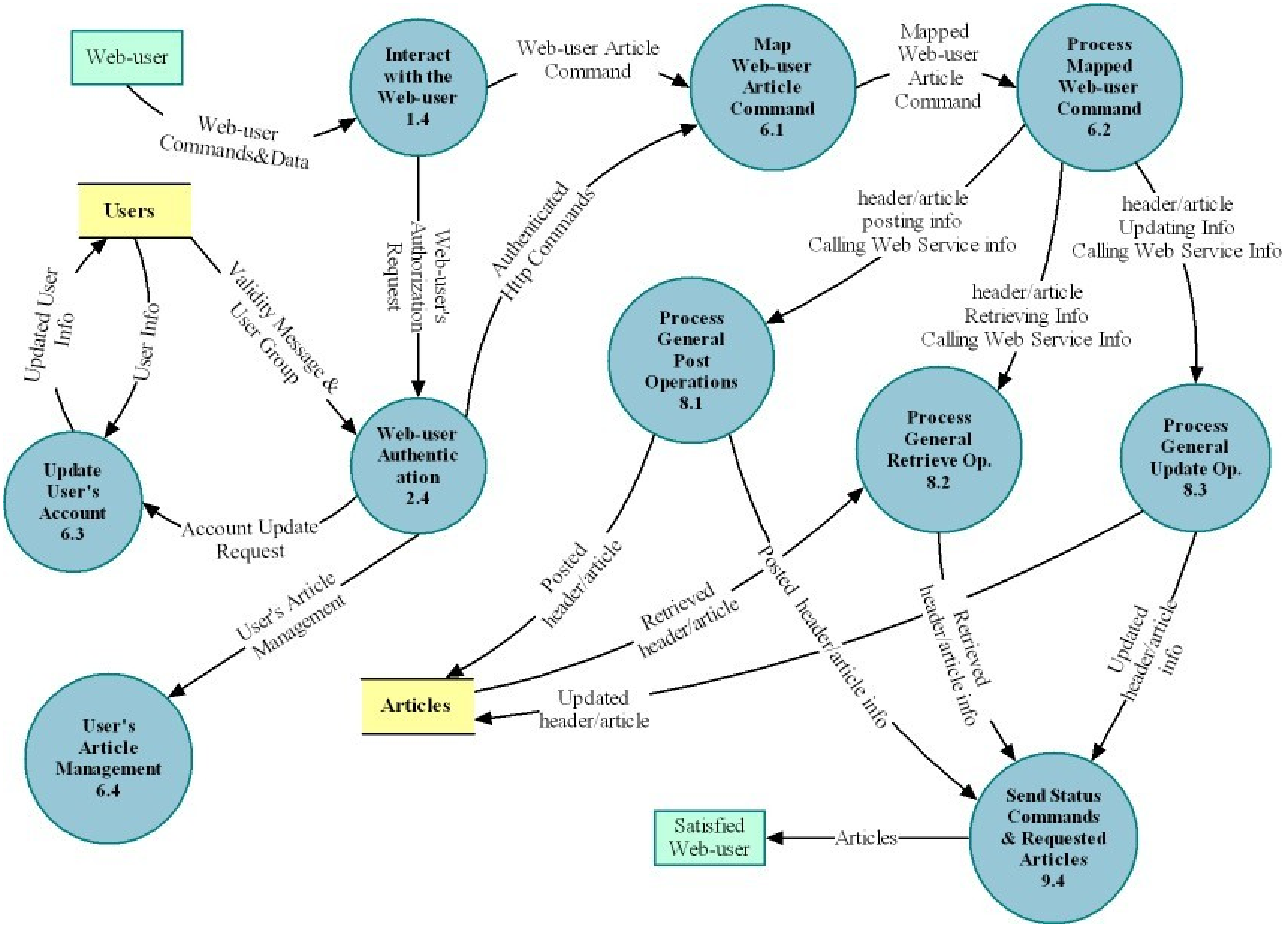


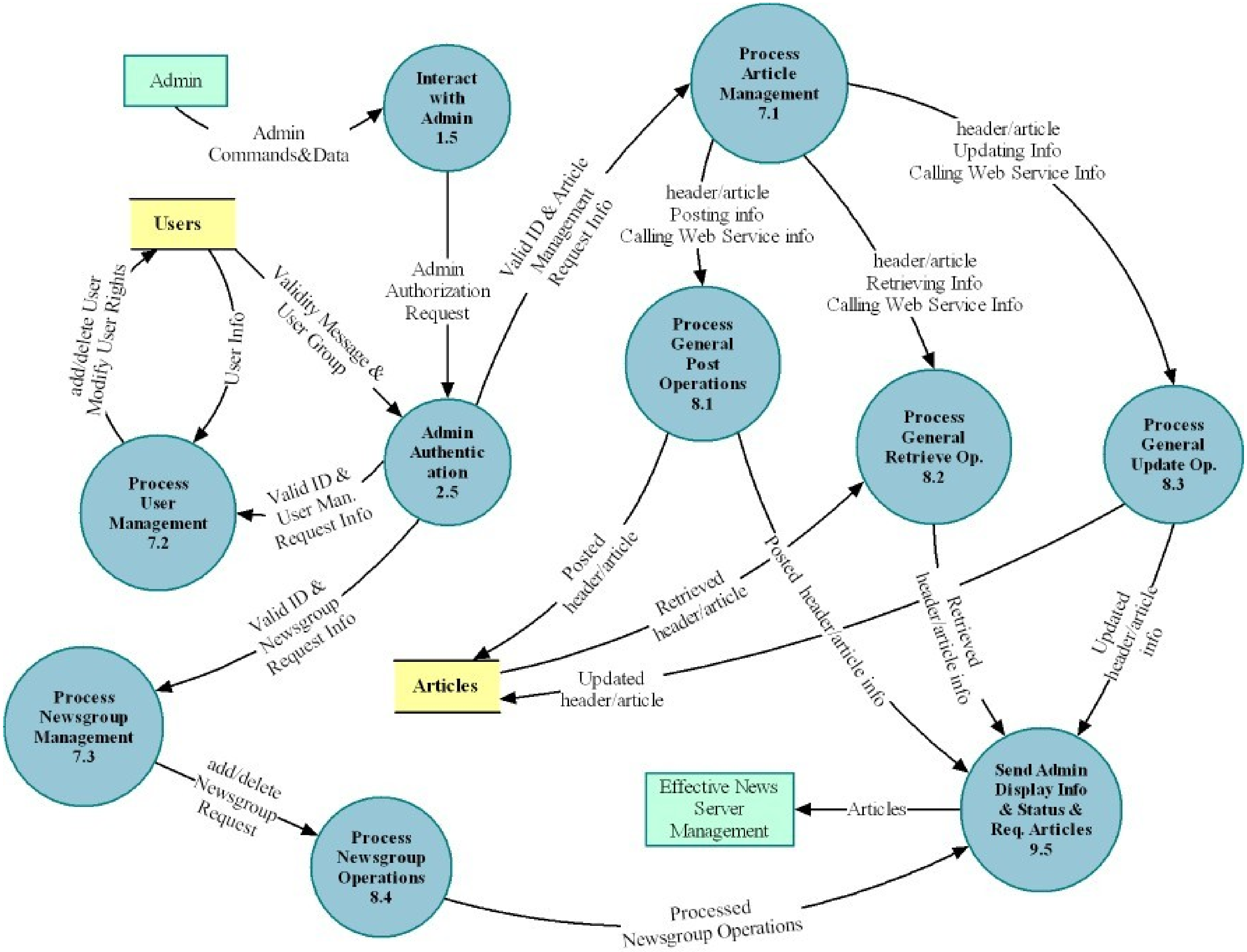
5.2.1.3 Level 2 Data Flow Diagrams


NNTP Client
NNTP Client Commands&Data
Interact with the NNTP Client 1.1
NNTP read Command
Map the NNTP Command 3.1
Mapped Command Info
Process Mapped NNTP Command 3.2
Users
Request
Authorization
NNTP-user's
Authenticated NNTP Commands
Validity Message & User Group
NNTP-user's Authentication 2.1
header/article posting info Calling Web Service info
header/article Updating Info Calling Web Service Info
header/article Retrieving Info Calling WebService Info
Process General Post Operations 8.1
Process General Retrieve Op. 8.2
Process General Update Op. 8.3
Posted header/article
Retrieved header/article
Updated header/article
Posted header/article info
Retrieved header/article
Updated header/article info
Articles
Satisfied NNTP-user
Articles
Send Status Commands & Requested Articles 9.1


RSS/Atom Client (Reader)
Commands & Data
Interact with the RSS/Atom Client 1.2
RSS/Atom Protocol Commands
Map RSS/Atom Command 4.1
Users
RSS/Atom user's Authentication Request
Authenticated RSS/Atom Protocol Commands
Mapped Command Info
Validity Message & User Group
RSS/Atom User Authen-tication 2.2
Process Mapped RSS/Atom command 4.2
Articles
Retrieved header/article
Process General Retrieve Op. 8.2
header/article Retrieving Info Calling WebService Info
Retrieved header/article
Send Requested RSS/Atom Feeds 9.2
RSS/Atom Feeds
Effective Feeding


Mail Client Commands&Data
Interact with Mail Client 1.3
Map the SMTP Command 5.1
Mapped Command Info
Process Mapped SMTP Command 5.2
Mail Client
Mail User's Authentication Request
Authenticated SMTP command & e-mail
header/article posting info Calling Web Service info
header/article Retrieving Info Calling Web Service Info
Validity Message & User Group
Mail User Authentic ation 2.3
Process General Post Operations 8.1
Process General Retrieve Op. 8.2
Users
Posted header/article
Retrieved header/article
Posted header/article info
Retrieved header/article info
Articles
Satisfied Mail User
Articles
Send Status Commands& Requested Articles Through POP3 9.3


Web-user
Web-user Commands&Data
Interact with the Web-user 1.4
Web-user Article Command
Map Web-user Article Command 6.1
Mapped Web-user Article Command
Process Mapped Web-user Command 6.2
Users
Updated User Info
User Info
Validity Message & User Group
Web-user's Authorization Request
Authenticated Http Commands
header/article posting info Calling Web Service info
header/article Updating Info Calling Web Service Info
header/article Retrieving Info Calling Web Service Info
Process General Post Operations 8.1
Process General Retrieve Op. 8.2
Process General Update Op. 8.3
Update User's Account 6.3
Web-user Authentic ation 2.4
Account Update Request
User's Article Management
Posted header/article
Retrieved header/article
Posted header/article info
Retrieved header/article info
Updated header/article info
Articles
Updated header/article
User's Article Management 6.4
Satisfied Web-user
Articles
Send Status Commands & Requested Articles 9.4


Admin
Admin Commands&Data
Interact with Admin 1.5
Users
add/delete User Modify User Rights
User Info
Validity Message & User Group
Admin Authorization Request
Valid ID & Article Management Request Info
Process Article Management 7.1
header/article Posting info Calling Web Service info
header/article Retrieving Info Calling Web Service Info
header/article Updating Info Calling Web Service Info
Process General Post Operations 8.1
Process General Retrieve Op. 8.2
Process General Update Op. 8.3
Admin Authentic ation 2.5
Process User Management 7.2
Valid ID & User Man. Request Info
Valid ID & Newsgroup Request Info
Posted header/article
Retrieved header/article
Posted header/article info
Retrieved header/article info
Updated header/article info
Articles
Updated header/article
Process Newsgroup Management 7.3
add/delete Newsgroup Request
Process Newsgroup Operations 8.4
Effective News Server Management
Articles
Send Admin Display Info & Status & Req. Articles 9.5
Processed Newsgroup Operations

5.2.2 Data Dictionary

Name:	NNTP Client Commands & Data
Aliases:	NNTP Requests
Where used/how used:	Interact with the NNTP Client 1.1 (Input) NNTP Client (Output)
Description: NNTP Client sends requests as in format stated in RFC-977. It also sends the required article information which is stated as data above.	

Name:	NNTP Read Command
Aliases:	NNTP Get Commands
Where used/how used:	Interact with the NNTP Client 1.1 (Output) Map the NNTP Command 3.1 (Input)
Description: After interaction with the NNTP Client without authentication, the client can only send NNTP get requests to get the header and article information.	

Name:	Mapped Command Info
Aliases:	None
Where used/how used:	Map the NNTP Command 3.1 (Output) Process Mapped NNTP Command 3.2 (Input)
Description: This data specifies the NNTP command to be processed which is mapped to corresponding function calls in NewsAgent system.	

Name:	header/article Updating Info
Aliases:	Web Service (WS) calling info for update
Where used/how used:	Process Mapped NNTP Command 3.2 (Output) Process General Update Op. 8.3 (Input) Process Mapped Web-user Command 6.2 (Output) Process Article Management 7.1 (Output)
Description: This data is the result of the mapping operation. It carries the required information to call XML web service for update operations of articles/headers from articles database.	
Name:	header/article Retrieving Info

Aliases:	Web Service (WS) calling info for retrieve
Where used/how used:	Process Mapped NNTP Command 3.2 (Output) Process Mapped RSS/Atom Command 4.2 (Output) Process Mapped SMTP Command 5.2 (Output) Process General Retrieve Op. 8.2 (Input) Process Mapped Web-user Command 6.2 (Output) Process Article Management 7.1 (Output)
Description: This data is the result of the mapping operation. It carries the required information to call XML web service for retrieve operations of articles/headers from articles database.	

Name:	header/article posting info
Aliases:	Web Service (WS) calling info for post
Where used/how used:	Process Mapped NNTP Command 3.2 (Output) Process Mapped SMTP Command 5.2 (Output) Process General Post Op. 8.1 (Input) Process Mapped Web-user Command 6.2 (Output) Process Article Management 7.1 (Output)
Description: This data is the result of the mapping operation. It carries the required information to call XML web service for posting operations of articles/headers from articles database.	

Name:	Authenticated NNTP Commands
Aliases:	Authenticated NNTP Requests
Where used/how used:	NNTP-user's Authentication 2.1 (Output) Map the NNTP Command 3.1 (Input)
Description: Authenticated NNTP Commands include all post, read, update etc. commands that an authenticated user may send.	

Name:	NNTP-user's Authorization Request
Aliases:	None
Where used/how used:	Interact with the NNTP Client 1.1 (Output) NNTP-user's Authentication 2.1 (Input)
Description: NNTP-user requests authorization in order to be an authenticated client. This data carries the required username password information.	

Name:	Validity Message & UserGroup
Aliases:	None
Where used/how used:	Users (Output) NNTP-user’s Authentication 2.1 (Input)
Description: As a result of the client’s authorization request validity message and the related user group is retrieved from the user information in the database.	

Name:	Posted header/article
Aliases:	None
Where used/how used:	Articles (Input) Process General Post Op. 8.1 (Output)
Description: This data is the new posted article information that will be inserted into the server database.	

Name:	Posted header/article Info
Aliases:	None
Where used/how used:	Send Status Commands & Requested Articles 9.1 (Input) Send Status Commands & Requested Articles Through POP3 9.3 (Input) Process General Post Op. 8.1 (Output) Send Status Commands & Requested Articles 9.4(Input) Send Admin Display Info Status & Requested Articles 9.5(Input)
Description: Posted header/article feedback info is sent back to the satisfied NNTP-user that tells the client that the post operation is successful.	

Name:	Retrieved header/article
Aliases:	None
Where used/how used:	Articles (Output) Process General Retrieve Op. 8.2 (Input)
Description: This data is the retrieved article/header information coming from the server database.	

Name:	Retrieved header/article Info
Aliases:	None
Where used/how used:	Send Status Commands & Requested Articles 9.1 (Input) Send Requested RSS/Atom feeds 9.2 (Input) Send Status Commands & Requested Articles Through POP3 9.3 (Input) Send Status Commands & Requested Articles 9.4(Input) Process General Retrieve Op. 8.2 (Output)
Description: Retrieved header/article info is sent back to the satisfied NNTP-user. It carries requested articles and header by the users.	

Name:	Updated header/article
Aliases:	None
Where used/how used:	Articles (Input) Process General Update Op. 8.3 (Output)
Description: This data is the updated article/header information that will be inserted to the server database.	

Name:	Updated header/article Info
Aliases:	None
Where used/how used:	Send Status Commands & Requested Articles 9.1 (Input) Process General Update Op. 8.3 (Output) Send Status Commands & Requested Articles 9.4 (Input)
Description: This data is the updated article/header feedback information that will be sent to the satisfied NNTP user.	

Name:	Articles
Aliases:	None
Where used/how used:	Send Status Commands & Requested Articles 9.1 (Output) Satisfied NNTP-user (Input) Satisfied Mail User (Input) Satisfied Web-user (Input) Effective News Server Management (Input)

Description: Article information that will be sent to the NNTP client.

Name:	RSS/Atom User's Commands & Data
Aliases:	None
Where used/how used:	Interact with the RSS/Atom Client 1.2 (Input) RSS/Atom Client (Output)
Description: This data includes the commands and data received through the RSS/Atom channel.	

Name:	RSS/Atom Protocol Commands
Aliases:	None
Where used/how used:	Interact with the RSS/Atom Client 1.2 (Output) Map RSS/Atom Command 4.1 (Input)
Description: RSS/Atom Client request commands coming through the unauthorized channel.	

Name:	RSS/Atom user's Authentication request
Aliases:	None
Where used/how used:	Interact with the RSS/Atom Client 1.2 (Output) RSS/Atom User Authentication 2.2 (Input)
Description: RSS/Atom client may request to be authenticated and this request is processed in RSS/Atom User Authentication Process. NewsAgent will use the Http authentication mechanism to authenticate the RSS/Atom client.	

Name:	Authenticated RSS/Atom Protocol Commands
Aliases:	None
Where used/how used:	Map RSS/Atom Command 4.1 (Input) RSS/Atom User Authentication 2.2 (Output)
Description: After RSS/Atom client to be authenticated, the client requests authenticated commands and these commands will be mapped into the appropriate calling web services methods.	

Name:	RSS/Atom Feeds
Aliases:	None
Where used/how used:	Send Requested RSS/Atom Feeds 9.2 (Output) Effective Feeding (Input)
Description: These feeds are provided for the RSS/Atom clients in order to retrieve data from our system. These feeds are in the format that is described in RSS 2.0 and Atom protocol formats.	

Name:	Validity Message & User Group
Aliases:	None
Where used/how used:	Users (Output) RSS/Atom User Authentication 2.2 (Input)
Description: After Authentication request, validity message and related user group info is retrieved from database in order to be processed in RSS/Atom User Authentication Process.	

Name:	Mapped Command Info
Aliases:	None
Where used/how used:	Map RSS/Atom Command 4.1 (Output) Process Mapped RSS/Atom Command 4.2 (Input)
Description: This information maps the related RSS/Atom Command info and processed in Mapped RSS/Atom Command process.	

Name:	Mail Client Commands & Data
Aliases:	None
Where used/how used:	Mail Client (Output) Interact with Mail Client 1.3 (Input)
Description: Mail client sends commands and data to the system.	

Name:	Mapped command Info
Aliases:	None
Where used/how used:	Map the SMTP Command 5.1 (Output) Process Mapped SMTP Command 5.2 (Input)

Description: Commands and data are mapped into related functions to call the web services and processed.

Name:	Mail User’s Authentication Request
Aliases:	None
Where used/how used:	Interact with Mail Client 1.3 (Output) Mail User Authentication 2.3 (Input)
Description: Mail user will get authenticated by his/her mail address by the system.	

Name:	Authenticated SMTP command & e-mail
Aliases:	None
Where used/how used:	Mail User Authentication 2.3 (Output) Map the SMTP Command 5.1 (Input)
Description: Mail clients are authenticated users because they provide e-mail addresses after signing up to the system. They can send electronic mails to the newsgroups and these will be converted to the appropriate articles by NewsAgent.	

Name:	Web-user Commands & Data
Aliases:	None
Where used/how used:	Web-user (Output) Interact with the Web-user 1.4 (Input)
Description: Web-user sends commands and data.	

Name:	Web-user Article Command
Aliases:	None
Where used/how used:	Interact with the Web-user 1.4 (Output) Map Web-User Article Command 6.1 (Input)
Description: Without authentication, Web-user only requests to read articles to the groups that unauthenticated users can subscribe.	

Name:	Mapped Web-user Article Command
Aliases:	None
Where used/how used:	Process Mapped Web-user Command 6.2 (Input) Map Web_user Article Command 6.1 (Output)
Description: Web-user's request to read article command is mapped to the related web service calling method.	

Name:	Web-user's Authorization Request
Aliases:	None
Where used/how used:	Interact with the Web_user 1.4 (Output) Web_user Authentication 2.4 (Input)
Description: Web_user may request to be an authenticated user and after authentication, web_user may request other commands.	

Name:	Authenticated HTTP Commands
Aliases:	None
Where used/how used:	Map Web_user Article Command 6.1 (Input) Web_user Authentication 2.4 (Output)
Description: After authentication, web-user will be able to request other commands including posting new articles, updating and deleting them, and these commands will be mapped to the related web services.	

Name:	Updated User Info
Aliases:	None
Where used/how used:	Users (Input) Update User's Account 6.3 (Output)
Description: After user info processed and updated, updated version of the user info is inserted to the database.	

Name:	User Info
Aliases:	None
Where used/how used:	Users (Output) Update User's Account 6.3 (Input) Process User Management 7.2 (Input)

Description:

User info is retrieved from the database and processed in the update user's account process.

Name:	Account Update Request
Aliases:	None
Where used/how used:	Web-user Authentication 2.4 (Output) Update User's Account 6.3 (Input)

Description:

An authenticated user will have the right of requesting account update and will be able to change personal info, e-mail, password etc.

Name:	User Article Management
Aliases:	None
Where used/how used:	Web-user Authentication 2.4 (Output) User's Article Management 6.4 (Input)

Description:

After authentication, user may request to manage user articles.

Name:	Admin's commands & data
Aliases:	None
Where used/how used:	Admin (Output) Interact with Admin 1.5 (Input)

Description:

Admin sends commands and data.

Name:	Admin Authorization Request
Aliases:	None
Where used/how used:	Interact with Admin (Output) Admin Authentication 2.5 (Input)

Description:

Admin requests to be authorized in order to run some administrative operations.

Name:	Valid ID & Article Man. Req. Info
Aliases:	None

Where used/how used:	Admin Authentication 2.5 (Output) Process Article Management 7.1 (Input)
Description: After admin authentication, admin requests to manage articles and this article management are processed. Validity ID also contains the access level of the admin. Some admin are not allowed to operate on critical operations.	

Name:	Valid ID & User Man. Req. Info
Aliases:	None
Where used/how used:	Admin Authentication 2.5 (Output) Process User Management 7.2 (Input)
Description: After admin authentication, admin requests to manage users and user management are processed. Validity ID also contains the access level of the admin. Some admin are not allowed to operate on critical operations.	

Name:	Valid ID & Newsgroup Man. Req. Info
Aliases:	None
Where used/how used:	Admin Authentication 2.5 (Output) Process Newsgroup Management 7.3 (Input)
Description: After admin authentication, admin requests to manage newsgroups and this newsgroup management are processed. Validity ID also contains the access level of the admin. Some admin are not allowed to operate on critical operations.	

Name:	Add/delete user Modify user rights
Aliases:	None
Where used/how used:	Users (Input) Process User Management 7.2 (Output)
Description: Admin may request to add, delete user or modify the users’ rights.	

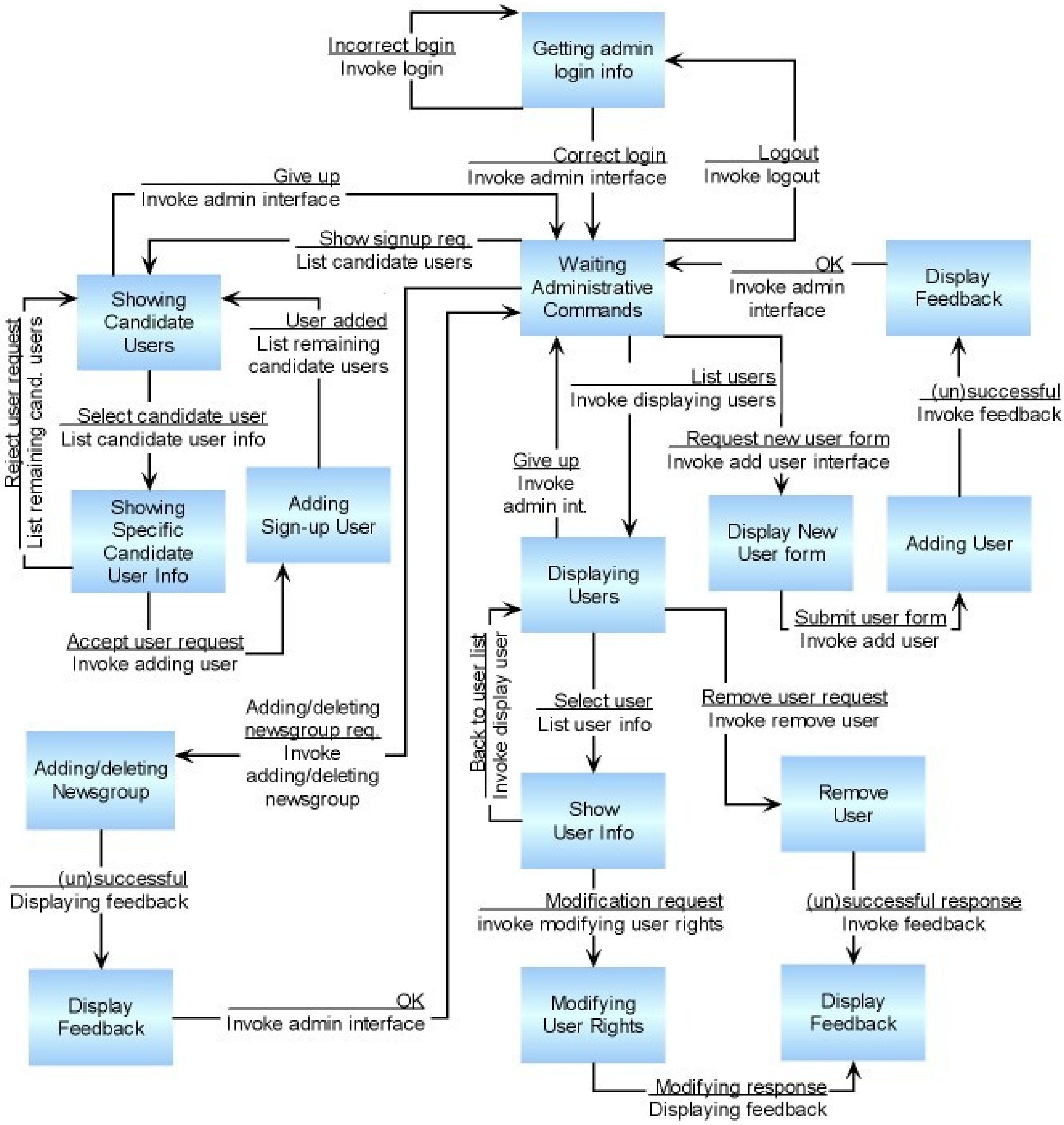
Name:	Add/delete Newsgroup Request
Aliases:	None
Where used/how used:	Process Newsgroup Op. 8.4 (Input) Process Newsgroup Management 7.3 (Output)

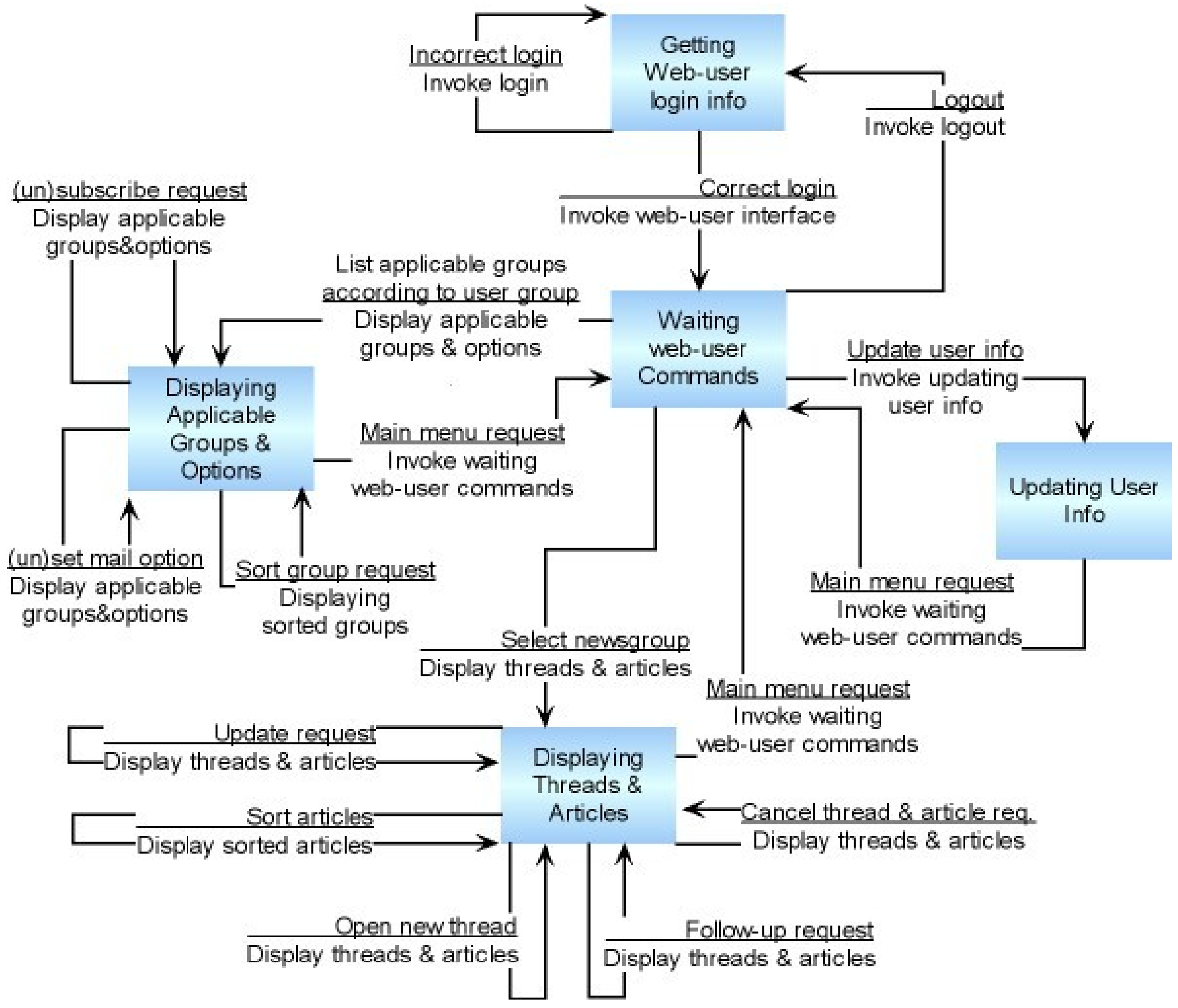
Description: Admin may request to add, delete newsgroup.

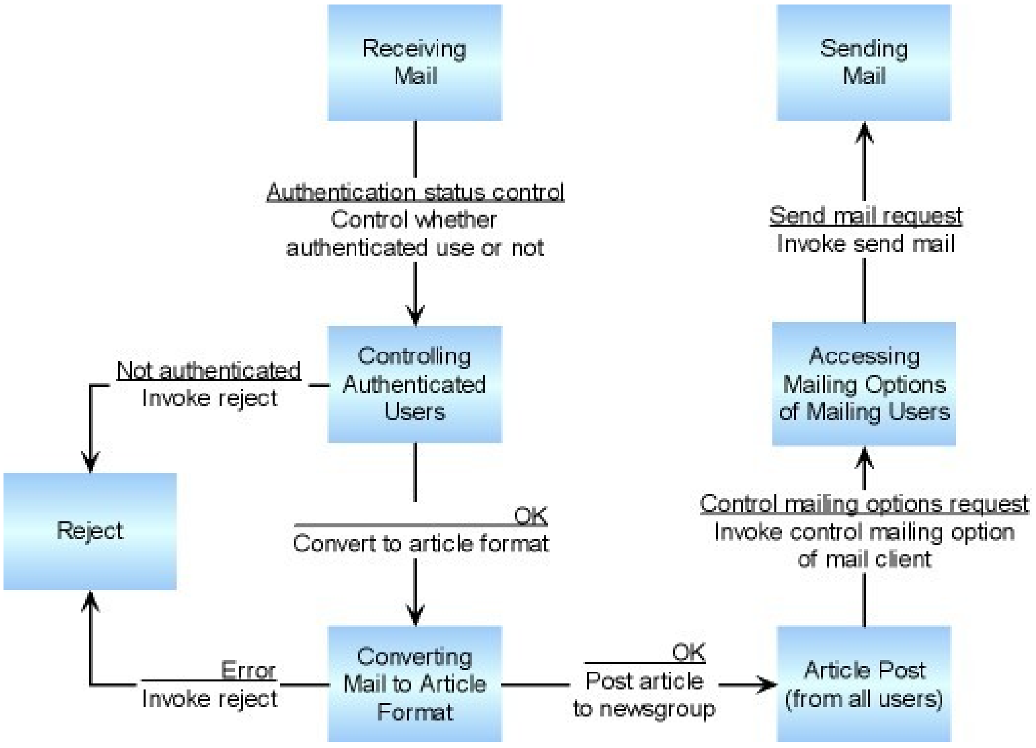
Name:	Processed Newsgroup Operations
Aliases:	None
Where used/how used:	Process Newsgroup Op. 8.4 (Input) Send Admin Display Info & Status & Req. Art. 9.5 (Input)
Description: Processed newsgroup operations are sent back to the admin.	

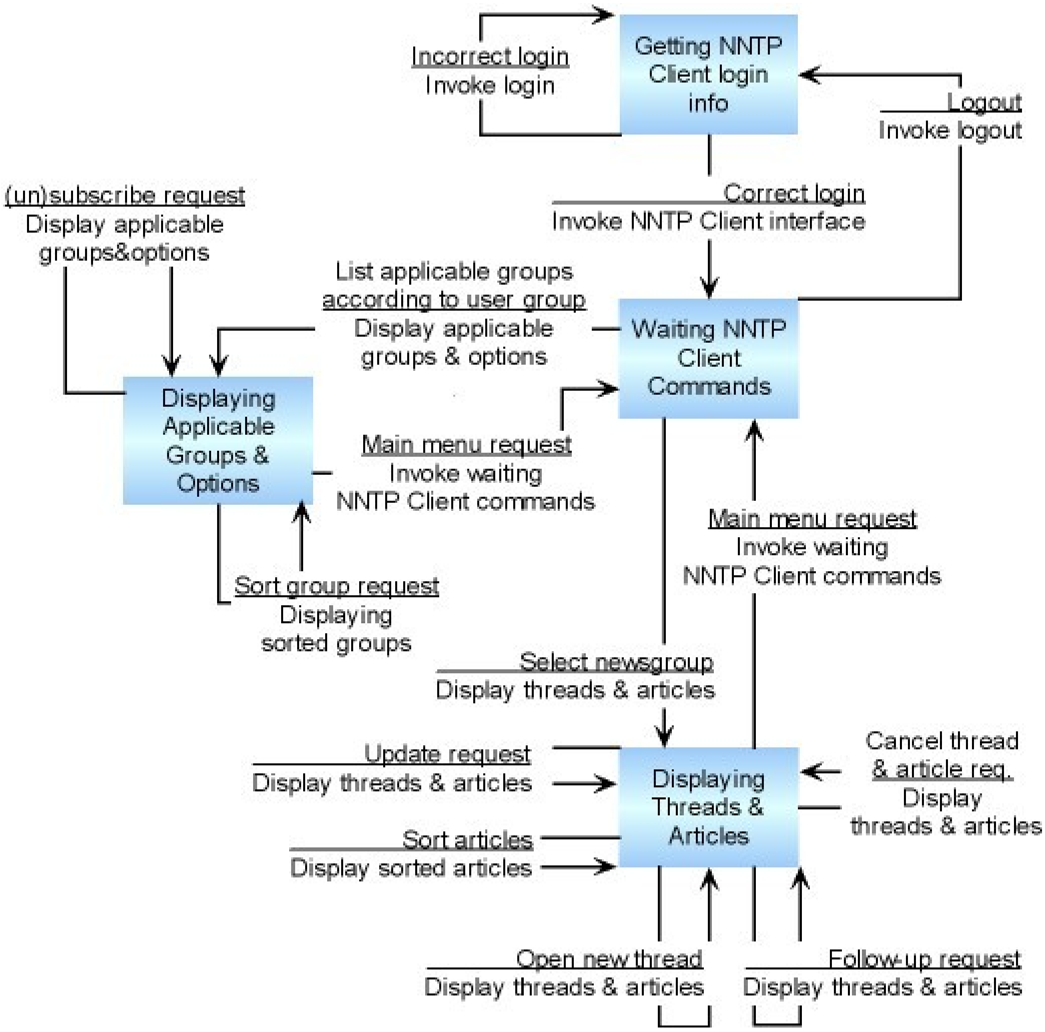
5.3 BEHAVIORAL MODELLING

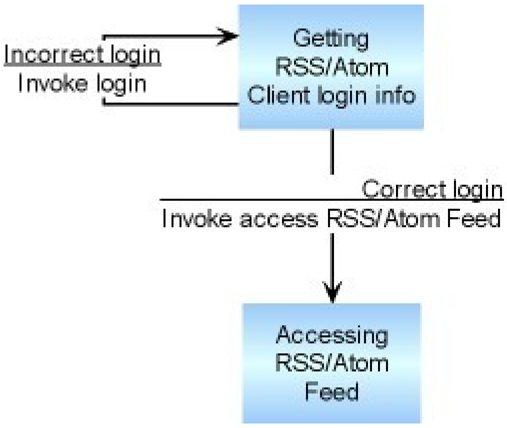
5.3.1 State Transition Diagrams


Incorrect login
Invoke login
Getting admin login info
Correct login
Invoke admin interface
Logout
Invoke logout
Give up
Invoke admin interface
Show signup req.
List candidate users
Waiting Administrative Commands
OK
Invoke admin interface
Display Feedback
Showing Candidate Users
User added
List remaining candidate users
Reject user request
List remaining cand. users
Select candidate user
List candidate user info
Showing Specific Candidate User Info
Adding Sign-up User
Accept user request
Invoke adding user
List users
Invoke displaying users
(un)successful
Invoke feedback
Give up
Invoke admin int.
Request new user form
Invoke add user interface
Display New User form
Adding User
Displaying Users
Submit user form
Invoke add user
Back to user list
Invoke display user
Select user
List user info
Remove user request
Invoke remove user
Adding/deleting newsgroup req.
Invoke adding/deleting newsgroup
Adding/deleting Newsgroup
Show User Info
Remove User
(un)successful
Displaying feedback
Modification request
invoke modifying user rights
(un)successful response
Invoke feedback
Display Feedback
OK
Invoke admin interface
Modifying User Rights
Display Feedback
Modifying response
Displaying feedback


Getting Web-user login info
Incorrect login
Invoke login
Logout
Invoke logout
Correct login
Invoke web-user interface
(un)subscribe request
Display applicable groups&options
List applicable groups according to user group
Display applicable groups & options
Waiting web-user Commands
Displaying Applicable Groups & Options
Update user info
Invoke updating user info
Updating User Info
Main menu request
Invoke waiting web-user commands
(un)set mail option
Display applicable groups&options
Sort group request
Displaying sorted groups
Main menu request
Invoke waiting web-user commands
Select newsgroup
Display threads & articles
Main menu request
Invoke waiting web-user commands
Update request
Display threads & articles
Displaying Threads & Articles
Sort articles
Display sorted articles
Cancel thread & article req.
Display threads & articles
Open new thread
Display threads & articles
Follow-up request
Display threads & articles


Receiving Mail
Authentication status control
Control whether authenticated use or not
Controlling Authenticated Users
Not authenticated
Invoke reject
Reject
OK
Convert to article format
Converting Mail to Article Format
Error
Invoke reject
OK
Post article to newsgroup
Article Post (from all users)
Control mailing options request
Invoke control mailing option of mail client
Accessing Mailing Options of Mailing Users
Send mail request
Invoke send mail
Sending Mail


Getting NNTP Client login info
Incorrect login
Invoke login
Logout
Invoke logout
Correct login
Invoke NNTP Client interface
(un)subscribe request
Display applicable groups&options
List applicable groups according to user group
Display applicable groups & options
Waiting NNTP Client Commands
Displaying Applicable Groups & Options
Main menu request
Invoke waiting NNTP Client commands
Main menu request
Invoke waiting NNTP Client commands
Sort group request
Displaying sorted groups
Select newsgroup
Display threads & articles
Update request
Display threads & articles
Displaying Threads & Articles
Cancel thread & article req.
Display threads & articles
Sort articles
Display sorted articles
Open new thread
Display threads & articles
Follow-up request
Display threads & articles


Incorrect login
Invoke login
Getting RSS/Atom Client login info
Correct login
Invoke access RSS/Atom Feed
Accessing RSS/Atom Feed

6 GANTT CHART

Gantt chart of NewsAgent is presented in APPENDIX A.

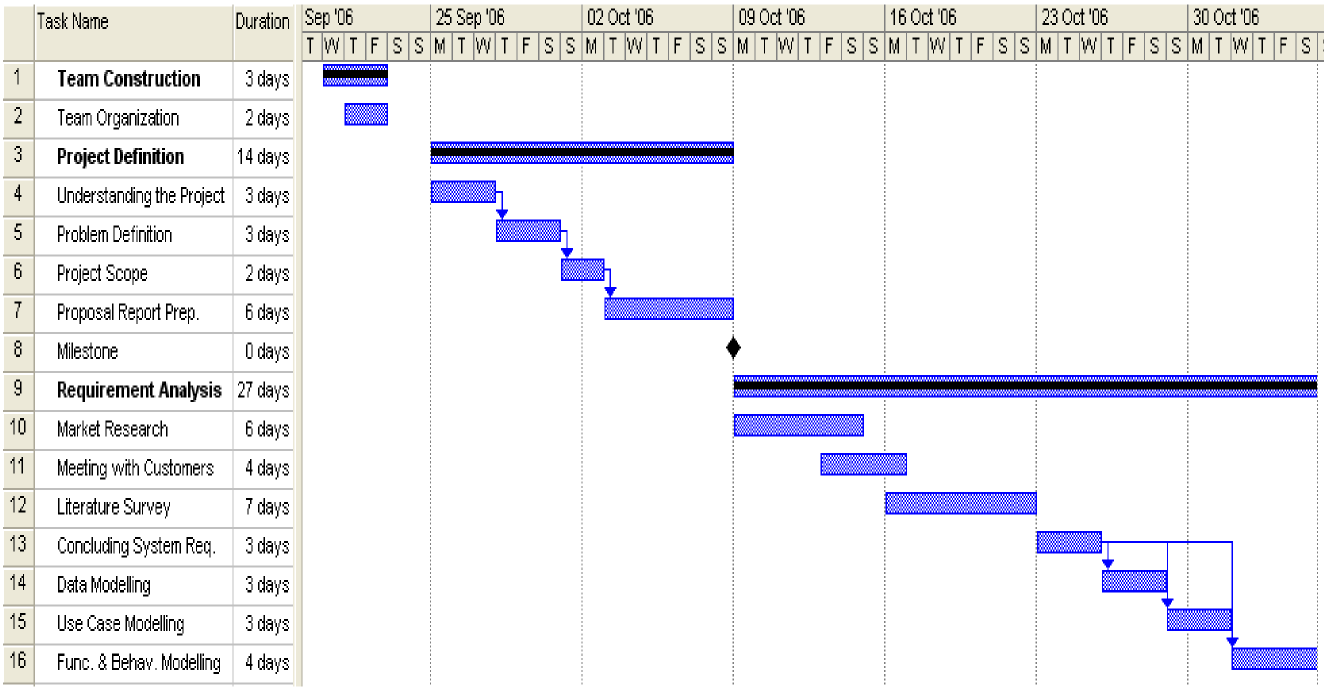
7 CONCLUSION

In today's technological platform, sharing information is one of the most important issues for both programmers and users. Therefore, developing existing systems, adding new features to them and designing new applications in this area are unavoidable requirements. In order to contribute to meeting these requirements, we have examined several similar projects about news servers. We specified the project progress by Gantt Chart.

From the beginning to the end, the whole process will be heavy-loaded and challenging, however, by the help of our coordination and team spirit we will easily come over these difficulties. We believe when NewsAgent takes its place in the market, users will easily realize the differences between earlier products and NewsAgent.

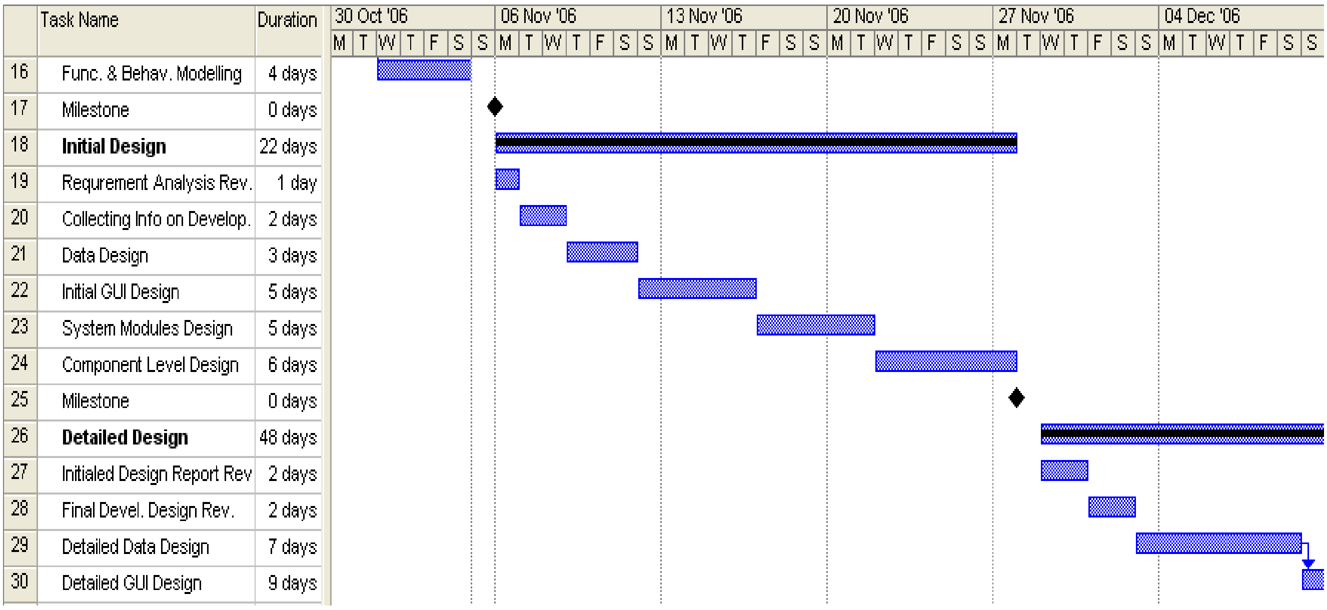
8 APPENDICES

8.1 APPENDIX A

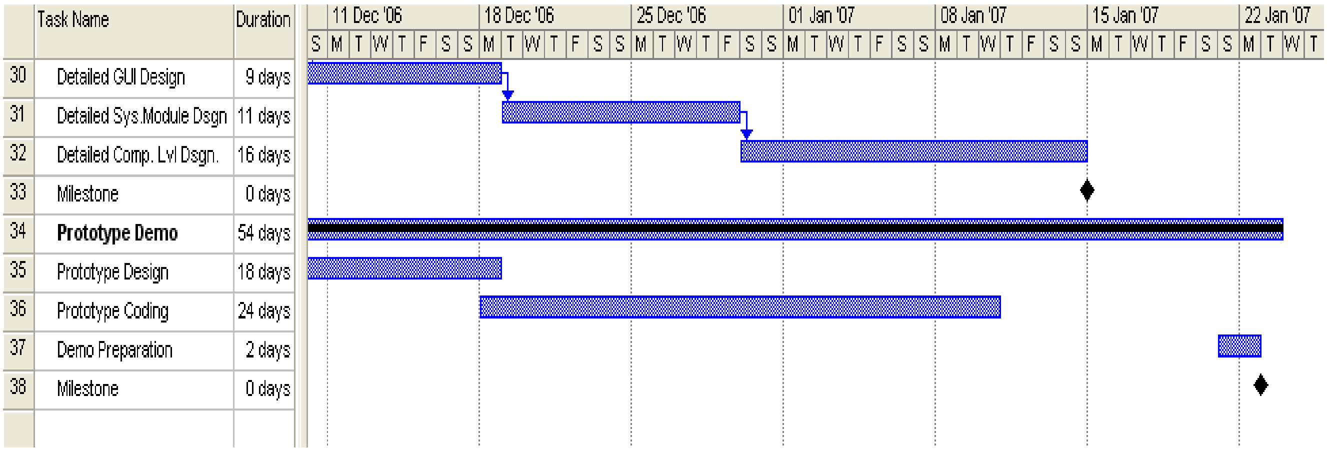


	Task Name	Duration
1	**Team Construction**	3 days
2	Team Organization	2 days
3	**Project Definition**	14 days
4	Understanding the Project	3 days
5	Problem Definition	3 days
6	Project Scope	2 days
7	Proposal Report Prep.	6 days
8	Milestone	0 days
9	**Requirement Analysis**	27 days
10	Market Research	6 days
11	Meeting with Customers	4 days
12	Literature Survey	7 days
13	Concluding System Req.	3 days
14	Data Modelling	3 days
15	Use Case Modelling	3 days
16	Func. & Behav. Modelling	4 days

Sep '06
25 Sep '06
02 Oct '06
09 Oct '06
16 Oct '06
23 Oct '06
30 Oct '06
T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S



	Task Name	Duration
16	Func. & Behav. Modelling	4 days
17	Milestone	0 days
18	**Initial Design**	22 days
19	Requrement Analysis Rev.	1 day
20	Collecting Info on Develop.	2 days
21	Data Design	3 days
22	Initial GUI Design	5 days
23	System Modules Design	5 days
24	Component Level Design	6 days
25	Milestone	0 days
26	**Detailed Design**	48 days
27	Initialed Design Report Rev	2 days
28	Final Devel. Design Rev.	2 days
29	Detailed Data Design	7 days
30	Detailed GUI Design	9 days



	Task Name	Duration
30	Detailed GUI Design	9 days
31	Detailed Sys.Module Dsgn	11 days
32	Detailed Comp. Lvl Dsgn.	16 days
33	Milestone	0 days
34	**Prototype Demo**	54 days
35	Prototype Design	18 days
36	Prototype Coding	24 days
37	Demo Preparation	2 days
38	Milestone	0 days

8.2 APPENDIX B

NNTP Base Commands

Command Code	**Command**	**Parameters**	**Description**
ARTICLE	Retrieve Article	Message ID or server article number.	Tells the server to send the client a particular USENET article. The article to be retrieved may be specified either using its absolute, universal message ID, or its locally-assigned article number. When the command is issued with an article number, this causes the server's

			internal message pointer to be set to the specified article. If the message pointer is already set to a particular article, the ARTICLE command can be issued without an article number and the current message will be retrieved.
HEAD	Retrieve Article Headers	Message ID or server article number.	Same as the ARTICLE command, but retrieves only the article's headers.
BODY	Retrieve Article Body	Message ID or server article number.	Same as the ARTICLE command, but returns only the body of the article.
STAT	Retrieve Article Statistics	Server article number	Conceptually the same as the ARTICLE command, but does not return any message text, only the message ID of the article. This command is usually used for the purpose of setting the server's internal message pointer, so STAT is normally invoked only with an article number (and not a message ID).
GROUP	Select Newsgroup	Newsgroup name	Tells the server the name of the newsgroup that the client wants to access. Assuming the group specified exists, the server returns to the client the numbers of the first and last articles currently in the group, along with an estimate of the number of messages in the group. The server's internal article pointer is also set to the first message in the group.
HELP	Get Help Information	None	Prompts the server to send the client help information, which usually takes the form of a list of valid commands that the server supports.
IHAVE	Offer Article To Server	Message ID	Used by the client in an NNTP session to tell the server that it has a new article that the server may want. The server will check the message ID provided and respond to the client indicating whether or not it wants the client to send the article.

LAST	Go To Last Message	None	Tells the server to set its current article pointer to the last message in the newsgroup.
LIST	List Newsgroups	None	Asks the server to send a list of the newsgroups that it supports, along with the first and last article number in each group. The command as described in RFC 977 is simple, supporting no parameters and causing the full list of newsgroups to be sent to the client. NNTP command extensions significantly expand the syntax of this command.
NEWGROUPS	List New Newsgroups	Date and time, and optional distribution specification	Prompts the server to send a list of new newsgroups created since the date and time specified. The client may also restrict the command to return only new newsgroups within a particular regional distribution.
NEWNEWS	List New News Articles	Date and time, and optional distribution specification	Requests a list from the server of all new articles that have arrived since a particular date and time. Like the NEWGROUPS command, this may be restricted in distribution. The server responds with a list of message IDs of new articles.
NEXT	Go To Next Message	None	Advances the server's current article pointer to the next message in the newsgroup.
POST	Post Article	None	Tells the server that the client would like to post a new article. The server responds with either a positive or negative acknowledgment. Assuming that posting is allowed, the client then sends the full text of the message to the server, which stores it and begins the process of propagating it to other servers.
SLAVE	Set Slave Status	None	This command is intended for use in special configurations where one NNTP server acts as a subsidiary to others. It is not often used in practice.

QUIT	End Session	None	Terminates the NNTP session. To be "polite", the client should issue this command prior to closing the TCP connection.

8.3 APPENDIX C

NNTP Status Responses and Response Codes

Similar to SMTP and FTP, NNTP commands can be considered as a three digit form (namely "xyz").

NNTP Reply Code Format: First Digit Interpretation
First Reply Code Digit ("x")

The first digit indicates the success, failure or progress of the command in general terms, whether a successful command is complete or incomplete, and the general reason why an unsuccessful command did not work. The values of this digit are defined slightly differently than in SMTP and FTP. In some cases, the terminology is just simplified; for example, the second category is "Command OK" instead of "Positive Completion Reply". Following table shows the specific meaning of the possible values of this digit:

NNTP Reply Code Format: Second Digit Interpretation
Second Reply Code Digit ("y")

Reply Code Format	**Meaning**	**Description**
1yz	**Informative Message**	General information; used for help information and debugging.
2yz	**Command OK**	The command was completed successfully.
3yz	**Command OK So Far, Send The Rest**	An intermediate reply, sent to prompt the client to send more information. Typically used for replies to commands such as *IHAVE* or *POST*, where the server acknowledges the command and then requests that an article be transmitted by the client.
4yz	**Command Was Correct, But Couldn't Be Performed**	The command was valid but could not be performed. This type of error usually occurs due to bad parameters, a transient problem with the server, bad command sequence or similar situations.
5yz	**Command Unimplemented Or Incorrect, Or Serious Program Error**	The command was invalid or a significant program error prevented it from being performed.

The second digit categorizes messages into functional groups. This digit is used in the same general way as in SMTP and FTP, but the functional groups are different; they are described in the following table.

Third Reply Code Digit (“z”)

This last digit indicates a specific type of message within each of the functional groups described by the second digit. The third digit allows each functional group to have 10 different reply codes for each reply type given by the first code digit.

Reply Code Format	Meaning	Description
x0z	**Connection, Setup and Miscellaneous**	Generic and miscellaneous replies.
x1z	**Newsgroup Selection**	Messages related to commands used to select a newsgroup.
x2z	**Article Selection**	Messages related to commands used to select an article.
x3z	**Distribution Functions**	Messages related to the transfer of messages.
x4z	**Posting**	Messages related to posting messages.
x5z	**Authentication**	Messages related to authentication and the *AUTHINFO* command extension. (This category is not officially listed in the standard, but these responses have a middle digit of “5”).
x8z	**Nonstandard Extensions**	Reserved for private, non-standard implementation use.
x9z	**Debugging**	Debugging output messages.

Combining Digit Values to Make Specific Reply Codes

As in FTP and SMTP, these “x”, “y” and “z” digit meanings are combined to make specific reply codes. For example, the reply code “435” is sent by the server if a client issues the *IHAVE* command but the server doesn't want the article being offered. The command was correct but the reply is negative, thus it starts with “4”, and the message is related to message distribution, so the middle digit is “3”.

8.4 APPENDIX D

NNTP Reply Codes

Reply Code	Meaning	Description
100	help text follows	Precedes response to *HELP* command.
111	(date and time)	Response to *DATE* command extension.
199	(debugging output)	Debugging information.
200	server ready - posting allowed	Sent by the server upon initiation of the session, if the client is allowed to post messages.
201	server ready - no posting allowed	Sent by the server upon initiation of the session, if the client is not allowed to post messages.
202	slave status noted	Response to the *SLAVE* command.
203	streaming is ok	Successful response to *MODE STREAM* command.
205	closing connection - goodbye!	Goodbye message sent in response to a *QUIT* message.
211	n f l s group selected	Successful response to the *GROUP* command, indicating the estimated number of messages in the group (“n”), first and last article numbers

		(“f” and “l”) and group name (“s”).
215	list of newsgroups follows (OR) information follows	Successful response to *LIST* command. The second form is for variations of *LIST* defined as NNTP command extensions
218	tin-style index follows	Successful response to *XINDEX* command extension.
220	n <a> article retrieved - head and body follow	Successful response to the *ARTICLE* command, indicating the article number and message ID of the article.
221	n <a> article retrieved - head follows	Successful response to the *HEAD* command, indicating the article number and message ID of the article.
222	n <a> article retrieved - body follows	Successful response to the *BODY* command, indicating the article number and message ID of the article.
223	n <a> article retrieved - request text separately	Successful response to the *STAT* command, indicating the article number and message ID of the article.
224	overview information follows	Successful response to the *XOVER* command extension.
230	list of new articles by message-id follows	Successful response to the *NEWNEWS* command.
235	article transferred ok	Successful response to the *IHAVE* command, after article has been sent.

239	article transferred ok	Successful response to the *TAKETHIS* command.
240	article posted ok	Successful response to the *POST* command, after article has been posted.
250 or 281	authentication accepted	Successful authentication using the *AUTHINFO* command extension.
282	list of groups and descriptions follows	Positive response to the *XGTITLE* command extension.
288	binary data to follow	Successful response to the *XTHREAD* command extension.
335	send article to be transferred	Preliminary response to the *IHAVE* command.
340	send article to be posted	Preliminary response to the *POST* command.
381	more authentication information required	Preliminary response to the *AUTHINFO* command extension.
400	service discontinued	Session is being terminated, perhaps due to user request.
411	no such newsgroup	Invalid newsgroup name specified.
412	no newsgroup has been selected	Attempt to issue a command that refers to the current newsgroup before one has been selected using *GROUP*.
420	no current article has been selected	Attempt to issue a command that refers to the current article using the server's current article

		pointer, before the pointer has been set through article selection.
421	no next article in this group	Response to *NEXT* command when at last article of a newsgroup.
422	no previous article in this group	Possible response to *LAST*; I have no idea why the word “previous” is in there.
423	no such article number in this group	Command with invalid article number.
430	no such article found	Article not found; it may have been deleted.
435	article not wanted - do not send it	Negative response to *IHAVE* if server doesn't need the article.
436	transfer failed - try again later	Temporary failure of article transfer, retry.
437	article rejected - do not try again	Article refused for whatever reason.
438	already have it, please don't send it to me	Same as reply code 435, but for the *CHECK* command extension.
440	posting not allowed	*POST* command issued when posting is not allowed.
441	posting failed	*POST* command failed.
450	authorization required for this command	Response sent when server requires authentication but client has not yet authenticated.

452	authorization rejected	Failed authentication.
480	transfer permission denied	Response to *CHECK* if transfer is not allowed.
500	command not recognized	Bad command.
501	command syntax error	Bad syntax in command.
502	access restriction or permission denied	Permission denied; sent if the client has not properly authentication but the server requires it.
503	program fault - command not performed	General fatal error message.

9 REFERENCES

1. http://www.infosys.tuwien.ac.at/NewsCache/doc/msthesis/NewsCache.html

2. http://www.mutantpenguin.co.uk/mpnews/features/

3. http://www.mibsoftware.com/userkt/inn/0003.htm

4. http://www.tcpipguide.com

4. http://en.wikipedia.org/wiki/MD5

5. http://www.ietf.org/rfc/rfc0850.txt