ServerTheon Project
Requirement Analysis Report

BY
TheonTech
TABLE OF CONTENTS

1.0 Project Scope and Description
   1.1 Problem Definition
   1.2 Project Scope
2.0 Market Observation
   2.1 Literature Survey
      2.1.1 Hamster Playground
      2.1.2 JAMES
      2.1.3 nntp//rss
3.0 Project Schedule
   3.1 Work Breakdown Structure
   3.2 Gantt Chart
4.0 Requirement Analysis
   4.1 System Requirements
      4.1.1 Software Requirements
      4.1.2 Hardware Requirements
      4.1.3 Tools
   4.2 Use-Case Analysis
      4.2.1 Use-Case Scenarios
      4.2.2 User Profiles
         4.2.2.1 Newsgroup Client
         4.2.2.2 Web Client
         4.2.2.3 Mail Client
         4.2.2.4 RSS Client
         4.2.2.4.2 Administrator
      4.2.3 Use-Case Diagrams
         4.2.3.1 The Use-Case of the Newsgroup Client
         4.2.3.2 The Use-Case of the Web Client
         4.2.3.3 The Use-Case of the Mail Client
         4.2.3.4 The Use-Case of the RSS Client
4.2.3.5 The Use-Case of the Administrator

5.0 Data Model and Description (E/R Diagrams)
   5.1 Data Objects
   5.2 Relationships
   5.3 Complete Data Model
   5.4 Data Dictionary

6.0 Functional Model and Description (Data Flow Diagrams)
   6.1 Level 0 Data Flow Diagram
   6.2 Level 1 Data Flow Diagram
   6.3 Level 2 Data Flow Diagrams
      6.3.1 Data Flow Diagram for Registration
      6.3.2 Data Flow Diagram for News Module
      6.3.3 Data Flow Diagram for Web Module
      6.3.4 Data Flow Diagram for Mail Module
      6.3.5 Data Flow Diagram for Core Service
   6.4 Data Dictionary of Data Flow Diagrams

7.0 Behavioral Model and Description (State Transition Diagram)

8.0 APPENDIX-A (Gantt Chart)

9.0 APPENDIX-B (Relationships)
1.0 Project Scope and Description

1.1 Problem Definition

Usenet is a set of protocols for generating, storing and retrieving news articles and for exchanging these articles among a readership which is potentially widely distributed. Usenet is one of the oldest computer network communications systems (established in 1980) still in widespread use.

Nowadays, the web forums and RSS newsfeeds are more commonly used for news broadcasting and discussion sessions. For the Unified News Exchange Server with NNTP, Mail, Web and RSS Project, we are expected to implement the followings:

- **The Message Exchange Core**: A basic threaded and secure message exchange service
- **The Extension Modules**: Modules providing e-mail lists, NNTP, WWW and RSS access methods

1.2 Project Scope

Whenever a subscriber posts a message to one of the extension modules, the following tasks will be accomplished:

- The article will also be posted as mail to the subscriber and to the related mail group.
- The article will also be posted to the related newsgroup.
- The article will also be fed to the RSS clients on web.
- The article will also be seen on the web forum.
2.0 Market Observation

2.1 Literature Survey

2.1.1 Hamster Playground

Hamster Playground is an open source unified mailserver and newsserver implemented under Delphi supporting NNTP, SMTP and POP3 protocols and SSL (see Figure-1).

Figure-1

Hamster Playground allows collecting news and mails from different servers. Collected messages are then available for any news reader or mail client on local computer or on any server in company network. Interesting feature of Hamster is that it could pull RSS feeds and parses them for being posted on newsgroups (see Figure-2).
Hamster Playground is split in two parts:

1. HService: HService is intended to run in the background and which does the actual work.
2. HControl: HControl contains the user interface and talks to HService over a TCP/IP connection.

The advantage is that HControl does not have to be installed on the same computer; it can be run from any computer, which can access the HService one by TCP/IP:
2.1.2 JAMES (Java Apache Mail Enterprise Server)

The Apache Java Enterprise Mail Server (James) is a Java SMTP and POP3 Mail server and NNTP News server. James has been designed to be a complete and portable enterprise mail engine solution based on currently available open protocols.

James is also a mail application platform. Mailet Java API is developed to let the developers write Java code to process emails. A mailet can generate an automatic reply, update a database, prevent spam and build a message archive. The James project hosts the Mailet API, and James provides an implementation of this mail application platform API.

<table>
<thead>
<tr>
<th>Item</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMTP server</td>
<td>Stable</td>
</tr>
<tr>
<td>Mailet Engine</td>
<td>Stable</td>
</tr>
<tr>
<td>FileSystem mailboxes/spool</td>
<td>Stable</td>
</tr>
<tr>
<td>RDBMS mailboxes/spool</td>
<td>Stable</td>
</tr>
<tr>
<td>POP3 server</td>
<td>Stable</td>
</tr>
<tr>
<td>RDBMS - Users</td>
<td>Stable</td>
</tr>
<tr>
<td>LDAP Support - Users</td>
<td>Experimental</td>
</tr>
<tr>
<td>TLS Support - POP3</td>
<td>Experimental</td>
</tr>
<tr>
<td>Remote Manager</td>
<td>Stable</td>
</tr>
<tr>
<td>TLS Support - Remote Manager</td>
<td>Stable</td>
</tr>
</tbody>
</table>
2.1.3 nntp//rss

nntp//rss is a Java-based bridge between RSS feeds and NNTP clients, enabling the client to read his/her favorite RSS syndicated content within his/her existing NNTP newsreader. RSS feeds are represented as NNTP newsgroups, providing a simple, straightforward and familiar environment for news reading.

nntp//rss contains both an NNTP server and an RSS aggregator, with a web interface for administration and monitoring. Installation is as simple as unpacking the distribution and running the application. It has been tested against popular NNTP newsreaders, including Mozilla, Outlook Express, MT-NewsWatcher, Free Agent, and text-based readers such as Gnus, tin and nn.

3.0 Project Schedule

3.1 Work Breakdown Structure (WBS)

1.0 Project Proposal
   1.1 Teaming-up and Determining Project Topic Preferences
   1.2 Determining Project Definition and Scope

2.0 System Requirements Specification
   2.1 Market Research
   2.2 Project Scheduling and Tracking
   2.3 Requirement Analysis
   2.4 Determination of Use-Cases
   2.5 Data Modeling & E/R Diagrams
   2.6 Functional Modeling & DFDS
2.7 Behavioral Modeling & State Transition Diagram
2.8 Data Dictionary

3.0 Initial Design
3.1 Data Design
3.2 Architectural Design
3.3 User-Interface Design
3.4 Component-Level Design
3.5 Testing Issues

4.0 Detailed Design
4.1 Design Review
4.2 Detailed Data Design
4.3 Detailed Architectural Design
4.4 Detailed User-Interface Design
4.5 Detailed Component-Level Design
4.6 Detailed Testing Issues

5.0 Prototype Development
5.1 Prototype Implementation
5.2 Prototype Demo

3.2 Gantt Chart
The Gantt Chart can be seen under Appendix-A.

4.0 Requirement Analysis
4.1 System Requirements
4.1.1 Software Requirements

| Server Side | - Unix / Linux
|             | - MySQL Server
|             | - Web Browser |
### Client Side
- Java Virtual Machine
- Unix / Linux / Windows
- Web Browser for forum access (optional)
- News Reader Application for newsgroup access (optional)
- RSS Aggregator for RSS access (optional)
- Mail Client Application for mailing list access (optional)

### Developers Side
- Eclipse Java IDE
- MySQL DBMS

#### 4.1.2 Hardware Requirements

| Server Side       | - Minimum 1024 MB RAM  
|                   | - A Pentium IV Processor  
|                   | - Minimum 10 GB Hard disk |
| Client Side       | - Minimum 256 MB RAM  
|                   | - A Pentium IV Processor  
|                   | - Minimum 5 GB Hard disk |
| Developers Side   | - Minimum 512 MB RAM  
|                   | - A Pentium IV Processor  
|                   | - Minimum 5 GB Hard disk |

#### 4.1.3 Development Tools
We will make use of the following tools during the development period of the project:
4.2.0 Use-Case Analysis

4.2.1 Usage Scenarios

The scenarios often called use-cases provide a description of how the system will be used. Once actors have been identified, use-cases can be developed. The use-case describes the manner in which an actor interacts with the system.

To create our use-cases; we have to identify the actors that use the ServerTheon. There are 6 actors in our system. These are Visitor, Newsgroup Client, Web Client, Mail Client, RSS Client and Administrator:

**Actor-1: Visitor**

**Usage Scenario -1:**

1. The visitor accesses our system via internet.
2. The visitor makes registration by using the sign-up feature of our system.
3. During registration, the visitor selects services which he/she wants.
4. After registration, the visitor logins by using the sign-in feature of our system.

**Actor-2: Newsgroup Client**
Usage Scenario -2:
1. The Newsgroup client subscribes to a group or groups which she/he wants to follow.
2. The Newsgroup client unsubscribes from a group or groups which she/he does not want to follow anymore.
3. The Newsgroup client reads messages in groups which he/she subscribed.
4. The Newsgroup client creates a new thread in a group which he/she subscribed.
5. The Newsgroup client sends a follow-up to a thread.
6. While replying a message, client quotes.
7. The Newsgroup client deletes his/her message.
8. The Newsgroup client logs out of our system.

Actor-3: Web Client

Usage Scenario -3:
1. The WebForum client reads a message in any group.
2. The WebForum client creates a new topic in any group.
3. The WebForum client replies to a message.
4. While replying a message, client quotes.
5. The WebForum client deletes or edits his/her message that he/she does not want to be read.
6. The WebForum client makes a search to find a keyword that he/she is looking for.
7. The WebForum client logs out of our system.

Actor-4: Mail Client

Usage Scenario -4:
1. The Mail client subscribes to a group or groups which she/he wants to follow.
2. The Mail client unsubscribes from a group or groups which she/he does not want to follow anymore.
3. The Mail client gets mail in groups which he/she subscribed.
4. The Mail client sends a mail to a group which he/she subscribed.
5. The Mail client replies to a mail.
6. While replying a message, client quotes.
7. The Mail client deletes his/her mail.
8. The Mail client logs out of our system.

**Actor-5: RSS Client**

**Usage Scenario -5:**
1. The RSS client subscribes to a group or groups which she/he wants to follow.
2. The RSS client unsubscribes from a group or groups which she/he does not want to follow anymore.
3. The RSS client gets mail in groups which he/she subscribed.
4. The RSS client sends a mail to a group which he/she subscribed.
5. The RSS client replies to a mail.
6. While replying a message, client quotes.
7. The RSS client deletes his/her mail.
8. The RSS client logs out of our system.

**Actor-6: Administrator (Admin)**

**Usage Scenario -6:**
1. Admin reads message in any group.
2. Admin sends message to any group when he/she wants to make an announcement or reply a message.
3. Admin deletes a message when he/she thinks that the message is inappropriate.
4. Admin edits the information of a client.
5. Admin deletes the account of a client.
6. Admin logs out of our system.

4.2.2 User Profiles

4.2.2.1 Newsgroup Client
- can subscribe to a group
- can unsubscribe from a group
- can read messages in groups
- can create a new thread in a group
- can send a follow-up to a thread
- can quote
- can delete his/her message
- can log out of the system

4.2.2.2 Web Client
- can read a message in any group
- can create a new topic in any group
- can reply to a message
- can quote
- can delete or edit his/her message
- can make a search to find a keyword
- can log out of the system

4.2.2.3 Mail Client
- can subscribe to a group
- can unsubscribe from a group
- can get mail from his/her groups
- can send a mail to one of his/her groups
- can reply to a mail
• can quote
• can delete his/her mail
• can log out of our system

4.2.2.4 RSS Client

• can subscribe to a group
• can unsubscribe from a group
• can get mail from his/her groups
• can send a mail to one of his/her groups
• can reply to a mail
• can quote
• can delete his/her mail
• can log out of our system

4.2.2.5 Administrator

• can read message in any group
• can send message to any group
• can delete inappropriate messages
• can edit the client information
• can delete the client account
• can log out of our system
4.2.3 Use-Case Diagrams

4.2.3.1 The Use-Case of the Newsgroup Client
4.2.3.2 The Use-Case of the Web Client

Visitor → Login → Web Client

Visitor → Post Messages

Visitor → Read Messages

Visitor → Delete Messages

Visitor → Edit Messages

Visitor → Search

Visitor → Logout
4.2.3.3 The Use-Case of the Mail Client

Visitor → Login → Mail Client

Mail Client → Send Mail → Get Mail → Delete Mail → Subscribe a Group → Unsubscribe a Group → Logout
4.2.3.4 The Use-Case of the RSS Client

4.2.3.5 The Use-Case of the Administrator
5.0 Data Model and Description (ER-diagrams)

5.1 Data Objects

a. Client
The Client entity saves all information related to a user. Client has to be authenticated. Some of the attributes are optional, i.e. can be ‘null’. Entity contains a primary key which is Client ID. Client ID is auto-incremented with seed one. Attributes are:

- ClientID
- ClientName
- ClientPassword
- Name
- Surname
- Gender
- DateOfBirth
- Address
- Country
- ZipCode
- Telephone
- AlternativeAddress
- AlternativeTelephone
- E-mail

b. Admin
The Admin entity saves all information related to an admin. Admin have to be authenticated. All attributes are mandatory, i.e. can not be ‘null’. Entity contains a primary key which is AdminID. AdminID is auto-incremented with seed one. Attributes are:
• AdminID
• AdminName
• AdminPassword
• Name
• Surname
• Gender
• DateOfBirth
• Address
• Country
• ZipCode
• Telephone
• AlternativeAddress
• AlternativeTelephone
• E-mail

c. Message
Message entity contains the information about messages. All attributes of the entity must be entered. However, ParentMessageID of first message will be null. Primary key is MessageID. MessageID is randomly generated by the system because of security reasons such as SQL injection. MessageSize is size of the message Attributes are:
• MessageID
• MessageText
• MessageSubject
• MessageSize
• ParentMessageID

d. Attachment
Attachment entity contains the information about attachments. All entries must be entered. Primary key is AttachmentID. AttachmentID is auto-incremented with seed one. Attributes are:

- AttachmentID
- FilePath
- FileName

e. Group

Group entity saves all information about groups such as computer is a group and hardware and software is its subgroups. All entries must be entered. Primary key is GroupID. GroupID is auto-incremented with seed one. Attributes are:

- GroupID
- GroupName
- GroupParentID

f. Service

Service entity saves all information about services. Services are news, mail, forum and RSS. All entries must be entered. Primary key is ServiceID. ServiceID is auto-incremented with seed one. Attributes are:

- ServiceID
- ServiceName

5.2 Relationships

a. Send

Each client may send zero or many messages. Every message must be sent by exactly one client. Send relation attributes are:

- MessageID
• ClientID
• DateTime

b. Select
Each client may select one or more services. Each service may be selected by zero or many client.
• ServiceID
• ClientID

c. Select1
Each client may select one or more groups. Each group may be selected by zero or many client.
• GroupID
• ClientID

d. Has
Every message has exactly one group. Every group may be owned by zero or many messages.
• MessageID
• GroupID

e. Has1
Each message has zero or many attachments. Every attachment must be owned by exactly one message.

f. Edit
Every message may be edited by zero or one client. Each client may edit zero or many message.
• MessageID
• ClientID

g. Delete
Every message may be deleted by zero or one client. Each client may delete zero or one message.
• MessageID
• ClientID

h. AdSend
Each admin may send zero or many messages. Each message is sent by only one admin
  • AdminID
  • MessageID

i. AdEdit
Each admin may edit zero or many messages. Each message may be edited by zero or many admin.
  • AdminID
  • MessageID

j. AdDelete
Each admin may delete zero or many messages. Each message may be deleted by zero or one admin.
  • AdminID
  • MessageID

k. AdEdit1
Each admin may edit zero or many client. Each client may be edited by zero or many admin.
  • AdminID
  • ClientID

l. AdDelete1
Each admin may delete zero or many client. Each client may be deleted by zero or one admin.
  • AdminID
  • ClientID
m. AdAdd2
Each admin must add one or more group. Each group must be added by exactly one admin.
- AdminID
- GroupID

n. AdEdit2
Each admin may edit zero or many group. Each group may be edited by zero or many admin.
- AdminID
- GroupID

o. AdDelete2
Each admin may delete zero or many group. Each group may be deleted by zero or one admin.
- AdminID
- GroupID

5.3 Complete Data Model
5.4 Data Dictionary

a. Client

<table>
<thead>
<tr>
<th>Data</th>
<th>Type- Size</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientID</td>
<td>int, auto-incr, not null</td>
<td>Number</td>
</tr>
<tr>
<td>ClientName</td>
<td>varchar(30), not null</td>
<td>Text</td>
</tr>
<tr>
<td>ClientPassword</td>
<td>varchar(10), not null</td>
<td>Hidden Text</td>
</tr>
<tr>
<td>Name</td>
<td>varchar(30), not-null</td>
<td>Text</td>
</tr>
<tr>
<td>Surname</td>
<td>varchar(30), not-null</td>
<td>Text</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------</td>
<td>------------</td>
</tr>
<tr>
<td>Gender</td>
<td>varchar(10), not-null</td>
<td>Text</td>
</tr>
<tr>
<td>DateOfBirth</td>
<td>Datetime</td>
<td>Date</td>
</tr>
<tr>
<td>Address</td>
<td>varchar(60)</td>
<td>Text</td>
</tr>
<tr>
<td>Country</td>
<td>varchar(20)</td>
<td>Text</td>
</tr>
<tr>
<td>ZipCode</td>
<td>Int</td>
<td>Number</td>
</tr>
<tr>
<td>Phone</td>
<td>varchar(15)</td>
<td>Text</td>
</tr>
<tr>
<td>AlternativeAddress</td>
<td>varchar(60)</td>
<td>Text</td>
</tr>
<tr>
<td>AlternativePhone</td>
<td>varchar(15)</td>
<td>Text</td>
</tr>
<tr>
<td>Email</td>
<td>varchar(40)</td>
<td>Text</td>
</tr>
</tbody>
</table>

### b. Admin

<table>
<thead>
<tr>
<th>Data</th>
<th>Type- Size</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>AdminID</td>
<td>int, auto-incr, not null</td>
<td>Number</td>
</tr>
<tr>
<td>AdminName</td>
<td>varchar(30), not-null</td>
<td>Text</td>
</tr>
<tr>
<td>AdminPassword</td>
<td>varchar(10), not-null</td>
<td>Hidden Text</td>
</tr>
<tr>
<td>Name</td>
<td>varchar(30), not-null</td>
<td>Text</td>
</tr>
<tr>
<td>Surname</td>
<td>varchar(30), not-null</td>
<td>Text</td>
</tr>
<tr>
<td>Gender</td>
<td>varchar(10), not-null</td>
<td>Text</td>
</tr>
<tr>
<td>DateOfBirth</td>
<td>Datetime</td>
<td>Date</td>
</tr>
<tr>
<td>Address</td>
<td>varchar(60)</td>
<td>Text</td>
</tr>
<tr>
<td>Data</td>
<td>Type-Size</td>
<td>Format</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------</td>
<td>--------</td>
</tr>
<tr>
<td>Country</td>
<td>varchar(20)</td>
<td>Text</td>
</tr>
<tr>
<td>ZipCode</td>
<td>Int</td>
<td>Number</td>
</tr>
<tr>
<td>Telephone</td>
<td>varchar(15)</td>
<td>Text</td>
</tr>
<tr>
<td>AlternativeAddress</td>
<td>varchar(60)</td>
<td>Text</td>
</tr>
<tr>
<td>AlternativeTelephone</td>
<td>varchar(15)</td>
<td>Text</td>
</tr>
<tr>
<td>E-mail</td>
<td>varchar(40)</td>
<td>Text</td>
</tr>
</tbody>
</table>

c. Message

<table>
<thead>
<tr>
<th>Data</th>
<th>Type-Size</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>MessageID</td>
<td>varchar(20)</td>
<td>Text</td>
</tr>
<tr>
<td>MessagText</td>
<td>varchar(100)</td>
<td>Text</td>
</tr>
<tr>
<td>MessageSubject</td>
<td>varchar(20)</td>
<td>Text</td>
</tr>
<tr>
<td>MessageSize</td>
<td>int</td>
<td>Number</td>
</tr>
<tr>
<td>ParentMessageID</td>
<td>varchar(20)</td>
<td>Text</td>
</tr>
</tbody>
</table>

d. Attachment

<table>
<thead>
<tr>
<th>Data</th>
<th>Type-Size</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>AttachmentID</td>
<td>int</td>
<td>Number</td>
</tr>
<tr>
<td>FilePath</td>
<td>varchar(100)</td>
<td>Text</td>
</tr>
<tr>
<td>FileName</td>
<td>varchar(50)</td>
<td>Text</td>
</tr>
</tbody>
</table>

e. Group

<table>
<thead>
<tr>
<th>Data</th>
<th>Type-Size</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>GroupID</td>
<td>int</td>
<td>Number</td>
</tr>
<tr>
<td>GroupName</td>
<td>varchar(50)</td>
<td>Text</td>
</tr>
<tr>
<td>GroupParentID</td>
<td>int</td>
<td>Number</td>
</tr>
</tbody>
</table>

f. Service

<table>
<thead>
<tr>
<th>Data</th>
<th>Type-Size</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>ServiceID</td>
<td>int</td>
<td>Number</td>
</tr>
</tbody>
</table>
6.0 Functional Model and Description (Data Flow Diagrams)

6.1 Level 0 Data Flow Diagram
6.2 Level 2 Data Flow Diagram

- Visitor
- Newsgroup Client
- Mail Client
- Web Client
- RSS Client
- DATABASE
- ADMIN

Diagram showing flow of data requests and responses between different modules and clients.
6.3 Level 2 Data Flow Diagrams

6.3.1 Data Flow Diagram for Registration

Diagram:
- Entry: sign-up request
- Processes:
  - Enter Registration Information
  - Enter Password
  - Preferences Selection
  - Confirm Registration
- Output:
  - member-info
- Flow:
  - registration information
  - password
  - preference.info
  - Selection Request
6.3.2 Data Flow Diagram for News Module

- NNTP Request
- News Request Handler
  - Authentication Process
    - Authentication request
    - Authentication result
- Message Operations Process
  - Post/Get/Delete requests
  - Service Response
  - Message request
- News Server
  - Other NNTP requests
  - News Data Request
  - NNIP Response
- Database
6.3.3 Data Flow Diagram for Web Module
6.3.4 Data Flow Diagram for Mail Module

- **SMTP Request**
- **Mail Request Handler**
  - e-mail message
  - Authentication request
  - Authentication result
- **E-mail Management Process**
  - Service Response
- **SMTP Server**
  - send mail request
- **Database**
  - Mail Data
  - Mail Data Request
  - SMTP Response
6.3.5 Data Flow Diagram for Core Service
### 6.4 Data Dictionary of Data Flow Diagrams

#### 1. NNTP Request

<table>
<thead>
<tr>
<th>Name:</th>
<th>NNTP Request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
</tbody>
</table>

**Where used / how used:**
News Request Handler (DFD Level 2: News Module) / Input given by Newsgroup Client

**Description:**
NNTP Request = request sent by client application via NNTP

#### 2. Post request

<table>
<thead>
<tr>
<th>Name:</th>
<th>Post request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
</tbody>
</table>

**Where used / how used:**
- News Request Handler (DFD Level 2: News Module) / Output

**Description:**
Post request = NNTP POST command

#### 3. Get request

<table>
<thead>
<tr>
<th>Name:</th>
<th>Get request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
</tbody>
</table>

**Where used / how used:**
- News Request Handler (DFD Level 2: News Module) / Output

**Description:**
Get request = NNTP ARTICLE command

#### 4. Delete request

<table>
<thead>
<tr>
<th>Name:</th>
<th>Delete request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
</tbody>
</table>

**Where used / how used:**
- News Request Handler (DFD Level 2: News Module) / Output
<table>
<thead>
<tr>
<th>Description:</th>
<th>Delete request = NNTP CANCEL command</th>
</tr>
</thead>
</table>

### 5. Other NNTP requests

<table>
<thead>
<tr>
<th>Name:</th>
<th>Other NNTP requests</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
</tbody>
</table>
| Where used / how used: | News Request Handler (DFD Level 2: News Module) / Output  
News Server (DFD Level 2: News Module) / Input |

| Description: | Other NNTP requests = other NNTP commands sent by news client application |

### 6. NNTP Response

<table>
<thead>
<tr>
<th>Name:</th>
<th>NNTP Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
</tbody>
</table>
| Where used / how used: | News Server (DFD Level 2: News Module) / Output  
sent to Newsgroup Client |

| Description: | NNTP Response = related NNTP response sent by news server |

### 7. Authentication request

<table>
<thead>
<tr>
<th>Name:</th>
<th>Authentication request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
</tbody>
</table>
| Where used / how used: | News Request Handler (DFD Level 2: News Module) / Output  
Authentication Process (DFD Level 2: News Module) / Input  
Mail Request Handler (DFD Level 2: Mail Module) / Output  
Authentication Process (DFD Level 2: Mail Module) / Input  
Web Request Handler (DFD Level 2: Web Module) / Output  
Authentication Process (DFD Level 2: Web Module) / Input |

| Description: | Authentication request = Tuple, compound object  
(username, password) |
8. Authentication result

<table>
<thead>
<tr>
<th>Name:</th>
<th>Authentication result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
</tbody>
</table>

Description:
Authentication result = [Valid | Invalid]

9. message request

<table>
<thead>
<tr>
<th>Name:</th>
<th>message request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
</tbody>
</table>

Description:
message request = [Post request | Get request | Delete request]

10. News Data Request

<table>
<thead>
<tr>
<th>Name:</th>
<th>News Data Request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
<tr>
<td>Where used / how used:</td>
<td>News Server (DFD Level 2: News Module) / Output to Database</td>
</tr>
</tbody>
</table>

Description:
News Data Request = SQL query sent to the database for related data operation

11. News Data
<table>
<thead>
<tr>
<th>Name:</th>
<th>News Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
<tr>
<td>Where used / how used:</td>
<td>News Server (DFD Level 2: News Module) / Input from Database</td>
</tr>
</tbody>
</table>

**Description:**

News Data = Result sent from the database for SQL query sent

News Data = [Article(s) | Group(s)]

### 12. Message

<table>
<thead>
<tr>
<th>Name:</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
</tbody>
</table>

**Description:**

Message = pre-defined message string passed to the core service

Message = [New Post Message | User Related Message | Group Related Message]

### 13. Service Response

<table>
<thead>
<tr>
<th>Name:</th>
<th>Service Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
<tr>
<td>Where used / how used:</td>
<td>Respond Process (DFD Level 2: Core Service) / Output to</td>
</tr>
</tbody>
</table>
News Module or Mail Module or Web Module
Message Operations Process (DFD Level 2: News Module) / Input from Core Service
E-mail Management Process (DFD Level 2: Mail Module) / Input from Core Service
Admin Operations Handler (DFD Level 2: Web Module) / Input from Core Service
Forum Operations Handler (DFD Level 2: Web Module) / Input from Core Service
RSS Operations Handler (DFD Level 2: Web Module) / Input from Core Service

**Description:**
Service Response = pre-defined response string returned by the core service

---

### 14. SMTP Request

<table>
<thead>
<tr>
<th>Name:</th>
<th>SMTP Request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
<tr>
<td>Where used / how used:</td>
<td>Mail Request Handler (DFD Level 2: Mail Module) / Input given by Mail Client</td>
</tr>
</tbody>
</table>

**Description:**
SMTP Request = request sent by sender-SMTP server via SMTP

---

### 15. SMTP Response

<table>
<thead>
<tr>
<th>Name:</th>
<th>SMTP Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
<tr>
<td>Where used / how used:</td>
<td>SMTP Server (DFD Level 2: Mail Module) / Output sent to Mail Client</td>
</tr>
</tbody>
</table>

**Description:**
SMTP Response = related SMTP response sent by SMTP server to the sender-SMTP server

---

### 16. e-mail message

<table>
<thead>
<tr>
<th>Name:</th>
<th>e-mail message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
<tr>
<td>Where used / how used:</td>
<td>Mail Request Handler (DFD Level 2: Mail Module) / Input given by Mail Client</td>
</tr>
</tbody>
</table>
### 17. send mail request

<table>
<thead>
<tr>
<th>Name:</th>
<th>Send mail request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
<tr>
<td>Where used / how used:</td>
<td>Mail Request Handler (DFD Level 2: Mail Module) / Output SMTP Server (DFD Level 2: Mail Module) / Input</td>
</tr>
</tbody>
</table>

**Description:**
send mail request = triggering signal for SMTP Server to deliver obtained e-mail to related receiver-SMTP servers

### 18. Mail Data Request

<table>
<thead>
<tr>
<th>Name:</th>
<th>Mail Data Request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
<tr>
<td>Where used / how used:</td>
<td>SMTP Server (DFD Level 2: Mail Module) / Output to Database</td>
</tr>
</tbody>
</table>

**Description:**
Mail Data Request = SQL query sent to the database for related data operation

### 19. Mail Data

<table>
<thead>
<tr>
<th>Name:</th>
<th>Mail Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
<tr>
<td>Where used / how used:</td>
<td>SMTP Server (DFD Level 2: Mail Module) / Input from Database</td>
</tr>
</tbody>
</table>

**Description:**
Mail Data = Result sent from the database in return for the SQL query sent

Mail Data = [E-mail message | E-mail address(es) of Mail list member(s)]
<table>
<thead>
<tr>
<th></th>
<th><strong>20. HTTP Request</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name:</strong></td>
<td>HTTP Request</td>
</tr>
<tr>
<td><strong>Aliases:</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Where used / how used:</strong></td>
<td>Web Request Handler (DFD Level 2: Web Module) / Input given by Web Client or RSS Client or Admin User</td>
</tr>
<tr>
<td><strong>Description:</strong></td>
<td>HTTP Request = request sent by client user’s web browser via HTTP</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th><strong>21. HTTP Response</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name:</strong></td>
<td>HTTP Response</td>
</tr>
<tr>
<td><strong>Aliases:</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Where used / how used:</strong></td>
<td>Web Server (DFD Level 2: Web Module) / Output sent to Web Client or RSS Client or Admin User</td>
</tr>
<tr>
<td><strong>Description:</strong></td>
<td>HTTP Response = related HTTP response sent by Web Server to the sender’s web browser</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th><strong>22. Admin Request</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name:</strong></td>
<td>Admin Request</td>
</tr>
<tr>
<td><strong>Aliases:</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Where used / how used:</strong></td>
<td>Web Request Handler (DFD Level 2: Web Module) / Output Admin Operations Handler (DFD Level 2: Web Module) / Input</td>
</tr>
<tr>
<td><strong>Description:</strong></td>
<td>Admin Request = a pre-defined parameter constructed from the request sent by admin user’s web browser via HTTP</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th><strong>23. Forum Request</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name:</strong></td>
<td>Forum Request</td>
</tr>
<tr>
<td><strong>Aliases:</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Where used / how used:</strong></td>
<td>Web Request Handler (DFD Level 2: Web Module) / Output Forum Operations Handler (DFD Level 2: Web Module) / Input</td>
</tr>
<tr>
<td><strong>Description:</strong></td>
<td>Forum Request = a pre-defined parameter constructed from the request sent by Forum Client’s web browser via HTTP</td>
</tr>
</tbody>
</table>
# 24. RSS Request

<table>
<thead>
<tr>
<th>Name:</th>
<th>RSS Request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
</tbody>
</table>
| Where used / how used: | Web Request Handler (DFD Level 2: Web Module) / Output  
RSS Operations Handler (DFD Level 2: Web Module) / Input |

**Description:**
RSS Request = a pre-defined parameter constructed from the request sent by RSS Clients’s web browser via HTTP

# 25. admin page request

<table>
<thead>
<tr>
<th>Name:</th>
<th>admin page request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
</tbody>
</table>
| Where used / how used: | Web Server (DFD Level 2: Web Module) / Input  
Admin Operations Handler (DFD Level 2: Web Module) / Output |

**Description:**
admin page request = triggering signal for Web Server to post constructed web page to Admin User’s web browser

admin page request = Tuple, compound object  
(trigger signal, page constructed)

# 26. forum page request

<table>
<thead>
<tr>
<th>Name:</th>
<th>forum page request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
</tbody>
</table>
| Where used / how used: | Web Server (DFD Level 2: Web Module) / Input  
Forum Operations Handler (DFD Level 2: Web Module) / Output |

**Description:**
forum page request = triggering signal for Web Server to post constructed web page to Forum Client’s web browser

forum page request = Tuple, compound object  
(trigger signal, page constructed)

# 27. RSS page request

<table>
<thead>
<tr>
<th>Name:</th>
<th>RSS page request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
<tr>
<td>Where used / how used:</td>
<td>Web Server (DFD Level 2: Web Module) / Input</td>
</tr>
</tbody>
</table>
### 28. Web Data Request

<table>
<thead>
<tr>
<th>Name:</th>
<th>Web Data Request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
<tr>
<td>Where used / how used:</td>
<td>Web Server (DFD Level 2: Web Module) / Output to Database</td>
</tr>
</tbody>
</table>

**Description:**
Web Data Request = SQL query sent to the database for related data operation

### 29. Web Data

<table>
<thead>
<tr>
<th>Name:</th>
<th>Web Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
<tr>
<td>Where used / how used:</td>
<td>Web Server (DFD Level 2: Web Module) / Input from Database</td>
</tr>
</tbody>
</table>

**Description:**
Web Data = Result sent from the database in return for the SQL query sent

Web Data = [Message(s) | Group(s) | RSS feeds(s) | User information]

### 30. New Post Message

<table>
<thead>
<tr>
<th>Name:</th>
<th>New Post Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
</tbody>
</table>
| Where used / how used: | Message Listener (DFD Level 2: Core Service) / Output  
New Post Handler (DFD Level 2: Core Service) / Input |

**Description:**
New Post Message = a pre-defined string for telling that a new post has been obtained

### 31. User Related Message

<table>
<thead>
<tr>
<th>Name:</th>
<th>User Related Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
</tbody>
</table>
32. Group Related Message

<table>
<thead>
<tr>
<th>Name:</th>
<th>Group Related Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
</tbody>
</table>

**Where used / how used:**
- Message Listener (DFD Level 2: Core Service) / Output
- Group Handler (DFD Level 2: Core Service) / Input

**Description:**
Group Related Message = a pre-defined string for telling that a group operation will be performed

33. Message Data

<table>
<thead>
<tr>
<th>Name:</th>
<th>Message Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
</tbody>
</table>

**Where used / how used:**
- Database Connector (DFD Level 2: Core Service) / Input & Output
- New Post Handler (DFD Level 2: Core Service) / Input & Output

**Description:**
Message Data = a pre-formatted text containing a message that is or will be stored in the database

Message Data = Tuple, compound object
- (MessageID, MessageText, MessageSubject, MessageSize, ParentMessageID)

34. User Data

<table>
<thead>
<tr>
<th>Name:</th>
<th>User Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
</tbody>
</table>

**Where used / how used:**
- Database Connector (DFD Level 2: Core Service) / Input & Output
- User Handler (DFD Level 2: Core Service) / Input & Output
### 35. Group Data

**Name:** Group Data  
**Aliases:** None  
**Where used / how used:** Database Connector (DFD Level 2: Core Service) / Input & Output  
Group Handler (DFD Level 2: Core Service) / Input & Output  

**Description:**  
Group Data = operation information text or group information text  

Group Data = [compound object (Operation Code, GroupID) | compound object (GroupID, GroupName, GroupParentID) ]

### 36. Operation Result

**Name:** Operation Result  
**Aliases:** None  
**Where used / how used:** New Post Handler (DFD Level 2: Core Service) / Output  
User Handler (DFD Level 2: Core Service) / Output  
Group Handler (DFD Level 2: Core Service) / Output  
Respond Process (DFD Level 2: Core Service) / Input  

**Description:**  
Operation result = Tuple, compound object  
([Success | Failure], Auxiliary Data {a pre-defined text related to operation performed} )

### 37. Core Data Request

**Name:** Core Data Request
<table>
<thead>
<tr>
<th><strong>Name</strong>:</th>
<th>Core Data</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Aliases</strong>:</td>
<td>None</td>
</tr>
<tr>
<td><strong>Where used / how used</strong>:</td>
<td>Database Connector (DFD Level 2: Core Service) / Input from Database</td>
</tr>
<tr>
<td><strong>Description</strong>:</td>
<td>Core Data = Result sent from the database in return for the SQL query sent</td>
</tr>
<tr>
<td></td>
<td>Core Data = [Message Data</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Name</strong>:</th>
<th>sign-up request</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Aliases</strong>:</td>
<td>None</td>
</tr>
<tr>
<td><strong>Where used / how used</strong>:</td>
<td>Enter Registration Information (DFD Level 2: Registration) / Input given by Visitor</td>
</tr>
<tr>
<td><strong>Description</strong>:</td>
<td>sign-up request = a string to inform the system about the new request</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Name</strong>:</th>
<th>registration information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Aliases</strong>:</td>
<td>None</td>
</tr>
<tr>
<td><strong>Where used / how used</strong>:</td>
<td>Enter Registration Information (DFD Level 2: Registration) / Output Enter Password (DFD Level 2: Registration) / Input</td>
</tr>
<tr>
<td><strong>Description</strong>:</td>
<td>registration information = Tuple, compound object (ClientName, Name, Surname, Gender, DateofBirth, Address, Country, ZipCode, Telephone, AlternativeAddress, AlternativeTelephone, E-mail)</td>
</tr>
</tbody>
</table>
### 41. Selection Request

<table>
<thead>
<tr>
<th>Name:</th>
<th>Selection Request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
<tr>
<td>Where used / how used:</td>
<td>Enter Registration Information (DFD Level 2: Registration) / Output Preference Selection (DFD Level 2: Registration) / Input</td>
</tr>
</tbody>
</table>

**Description:**
sign-up request = trigger for the system to send the page that the new user would select preferences on

### 42. Preference-info

<table>
<thead>
<tr>
<th>Name:</th>
<th>Preference-info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
<tr>
<td>Where used / how used:</td>
<td>Preference Selection (DFD Level 2: Registration) / Output Enter Password (DFD Level 2: Registration) / Input</td>
</tr>
</tbody>
</table>

**Description:**
Preference-info = Tuple, compound object (ServiceName(s), GroupName(s))

### 43. password

<table>
<thead>
<tr>
<th>Name:</th>
<th>password</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
<tr>
<td>Where used / how used:</td>
<td>Confirm Registration (DFD Level 2: Registration) / Input Enter Password (DFD Level 2: Registration) / Output</td>
</tr>
</tbody>
</table>

**Description:**
password = an encrypted string

### 44. member-info

<table>
<thead>
<tr>
<th>Name:</th>
<th>member-info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliases:</td>
<td>None</td>
</tr>
<tr>
<td>Where used / how used:</td>
<td>Confirm Registration (DFD Level 2: Registration) / Output to Database</td>
</tr>
</tbody>
</table>

**Description:**
member-info = SQL query for storing new member information into the database

member-info = Tuple, compound object
              (Registration information, password, Preference-info)
7.0 Behavioral Model and Description (State Transition Diagram)

- **Authentication state**
  - Authentication Successful & NNTP Request: send response & invoke news module
  - Authentication Failed: send response

- **Listening All state**
  - Any Request: start authentication

- **Listening HTTP state**
  - Authentication Successful & HTTP Request: send response & invoke news module

- **Listening SMTP state**
  - Authentication Successful & SMTP Request: send response & invoke news module

- **Listening NNTP state**
  - Authentication Successful & NNTP Request: send response & invoke news module

- **Core Processing state**
  - Processing Finished: send service response
  - Post/Get/Delete Request: invoke core service

- **Forum state**
  - Service Response: post page data
  - Post/Get/Delete Request: invoke core service

- **RSS state**
  - Processing Finished: send service response
  - Pull Request: send feed

- **Admin state**
  - Admin Request: invoke core service

- **Logout**
  - close news module
  - close web module
  - close mail module

- **Forum Request**
  - invoke forum handler

- **RSS Request**
  - invoke RSS handler

- **Admin Request**
  - invoke core service

- **Send Mail Request**
  - invoke core service

- **Processing Finished**
  - send service response
### 8.0 APPENDIX –A (Gantt Chart)

![Gantt Chart Image]

### 9.0 APPENDIX–B (Relationships)

![Relationships Diagram Image]