TEAM NAME : TOMBEKI++

PROJECT : WEBMES

ASSISTANT : CAGATAY CALLI

PREPARED BY:
M.ARIF UGUREL – 1348036
SERDAR SAYGILI – 1347897
BUGRA HASBEK – 1347541
RUSLAN ABDULLA – 1321751
TABLE OF CONTENTS

1 INTRODUCTION ............................................................................................................ 5
  1.1 PROJECT TITLE ........................................................................................................ 5
  1.2 PROBLEM DEFINITION & SCOPE ........................................................................ 5
    1.2.1 PROBLEM DEFINITION .............................................................................. 5
    1.2.2 PROJECT SCOPE ....................................................................................... 5
  1.3 DESIGN OBJECTIVES ............................................................................................. 7
    1.3.1 USABILITY .................................................................................................. 7
    1.3.2 RELIABILITY ............................................................................................. 7
    1.3.3 SECURITY .................................................................................................. 7
    1.3.4 PORTABILITY .............................................................................................. 7
    1.3.5 PERFORMANCE ......................................................................................... 7
2 DESIGN CONSTRAINTS ................................................................................................... 8
  2.1 SYSTEM CONSTRAINTS .......................................................................................... 8
    2.1.1 HARDWARE CONSTRAINTS ........................................................................ 8
    2.1.2 SOFTWARE CONSTRAINTS .......................................................................... 9
  2.2 TIME CONSTRAINTS ............................................................................................. 10
  2.3 PERFORMANCE CONSTRAINTS ............................................................................. 10
3 ARCHITECTURAL AND COMPONENT LEVEL DESIGN .............................................. 11
  3.1 MODULES ............................................................................................................. 11
    3.1.1 DISPLAYER MODULE .............................................................................. 11
    3.1.2 ANALYZER MODULE ............................................................................... 11
    3.1.3 MACHINE LEARNINGMODULE ............................................................... 11
    3.1.4 AUTHENTICATION MODULE ................................................................... 12
  3.2 DATA FLOW DIAGRAMS ........................................................................................ 13
    3.2.1 LEVEL 0 DFD ........................................................................................ 13
    3.2.2 LEVEL 1 DFD ........................................................................................ 14
    3.2.3 LEVEL 2 DFD ........................................................................................ 15
  3.2 STATE TRANSITION DIAGRAM ........................................................................... 31
4 OBJECT – ORIENTED DIAGRAMS .............................................................................. 32
  4.1 USE CASE DIAGRAM ........................................................................................... 32
  4.2 ACTIVITY DIAGRAMS .......................................................................................... 33
    4.2.1 REGISTERING .......................................................................................... 33
    4.2.2 LOGIN ..................................................................................................... 35
    4.2.3 ASKING QUESTION .................................................................................. 37
    4.2.4 QUESTION ANSWERING ....................................................................... 39
    4.2.5 CHATTING ................................................................................................. 41
    4.2.6 READING HELP ....................................................................................... 43
    4.2.7 PINNING A QUESTION ............................................................................ 45
4.2.8 REPORTING ABUSE ..................................................................................................................46
4.2.9 REPORTING BUG ......................................................................................................................47
4.3 CLASS DIAGRAMS .......................................................................................................................48
  4.3.1 AUTHENTICATION MODULE .................................................................................................48
  4.3.2 MACHINE LEARNING MODULE .............................................................................................49
  4.3.3 DISPLAYER MODULE ..............................................................................................................50
  4.3.4 ANALYZER MODULE ..............................................................................................................51
4.4 SEQUENCE DIAGRAMS ................................................................................................................52
  4.4.1 AUTHENTICATION MODULE .................................................................................................52
  4.4.2 MACHINE LEARNING MODULE .............................................................................................53
  4.4.3 DISPLAYER MODULE ..............................................................................................................54
  4.4.4 ANALYZER MODULE ..............................................................................................................55
5 USER INTERFACE DESIGN .............................................................................................................56
  5.1 INTRODUCTION ..........................................................................................................................56
  5.2 HOME ........................................................................................................................................56
  5.3 SEARCH ......................................................................................................................................58
  5.4 START CHAT ...............................................................................................................................59
  5.4 CHATTING ..................................................................................................................................60
  5.5 REGISTRATION ............................................................................................................................61
  5.6 HELP ..........................................................................................................................................62
  5.7 REPORT BUG ..............................................................................................................................63
  5.8 REPORT ABUSE ..........................................................................................................................64
  5.9 ABOUT MARPUCH ......................................................................................................................65
  5.10 ABOUT US ..............................................................................................................................66
6 PROCESS .........................................................................................................................................67
  6.1 TEAM STRUCTURE .....................................................................................................................67
  6.2 PROCESS MODEL .......................................................................................................................67
  6.3 GANNT CHART ..........................................................................................................................67
7 TESTING ...........................................................................................................................................68
  7.1 INTRODUCTION ..........................................................................................................................68
  7.2 TESTING STRATEGY ....................................................................................................................68
  7.3 UNIT TESTING .............................................................................................................................68
  7.4 INTEGRATION TESTING .............................................................................................................74
  7.5 VALIDATION TESTING ..............................................................................................................76
8 CONCLUSION ...................................................................................................................................77
9 REFERENCES ....................................................................................................................................78
10 APPENDIX .....................................................................................................................................79
1 INTRODUCTION

Internet has become very essential in our daily life. We use Internet for getting information on various subjects, sharing our videos and photos with friends and for finding people with similar interests. New trend in internet technology is Web 2.0 applications. In the old days, Internet had one-way information traffic. Internet sites were sending information but users could not contribute their knowledge or resources at all. Web 2.0 changed the face of internet with a genius approach and let the users to contribute to internet sites. Some famous Web 2.0 applications are wikipedia, youtube and facebook. These internet sites serve an environment to users and users add content to internet sites. Ajax is a technology used in Web 2.0 application which does not need to reload the entire page, it only reloads required parts. This feature increases the effectiveness of internet sites by shortening loading time.

1.1 PROJECT TITLE

Our company name is Tombeki++. We wanted to name our project in harmony with our company name so we decided to name our project Marpuch.

1.2 PROBLEM DEFINITION & SCOPE

1.2.1 PROBLEM DEFINITION

Internet has become information highway of the new world; however it is very easy to get lost while searching for information. You can find hordes of information on a subject however most of what you find will be totally useless or unrelated. Even if you find a useful site with related information, you will need to skim between many pages of tutorials to find information on a certain paragraph. This process wastes quite a time and also it will be quite boring. Moreover many users do not like/need generic answers. Users want solution for their very own, specific problems and they want it fast.

Furthermore, sometimes users need trivial information such as “the day with the longest night time” or “price of a certain Mercedes model”. Building a Web 2.0 site is the perfect answer for this problem.

1.2.2 PROJECT SCOPE

Marpuch will be an Ajax-based Web 2.0 application which answers users’ questions much faster than traditional ways.

Marpuch will provide users a question box. When a user asks a question in English language, Marpuch will analyze the question via natural language processing techniques. Then Marpuch will search its database for an appropriate answer. If the question is trivial, such as
"what is boiling point of water?" then answer will be shown. Otherwise we will show similar questions and their answers, we will talk about this later.

If no similar question is found or user is not satisfied with those questions then Marpuch will search online Marpuch volunteers who are expert in that field. This is the Web 2.0 side of Marpuch. As we described earlier, Web 2.0 applications take advantage of users' knowledge, Marpuch is no exception. We will allow users to 'tag' themselves as 'Expert' in a subject. As I said earlier, when a user cannot find a satisfactory answer in a certain field, Marpuch will search for volunteers who are expert in that subject. User will be able to talk with these experts via browser. When a user wants to talk with an expert, expert will be notified by his/her instant messenger software. This process will be done via Marpuch's jabber server. We think that user-expert inputs are extremely important so after an expert-user chat; experts will be able to mark some of user's sentences as 'the real question' and also will be able to mark his/her sentences as 'the real answer' of that question. These questions will be saved to Marpuch's database. On the other hand, we will ask registered users to rate expert with respect to our expert criteria. Moreover registered users will be able to comment on volunteers. Anyone with a browser will be able to use Marpuch without registering to system however we will only take feedback from registered users to prevent abuse of rating system. We will have a ladder system for volunteers (a.k.a. experts). User ratings, chat count and amount of online time will affect the total rating and class of volunteer.

We believe that if an expert chats with many users at the same time, his/her efficiency will be reduced so Marpuch will not let users to ask question to volunteers who are already in a chat with another user. However we will show volunteers who are in chat with another user in expert search as busy. These are the core elements of Marpuch.

If no expert is online when a question is asked, then registered users will be able to pin that question. When an expert on that subject becomes online, he/she will be notified about 'pinned question'. These are the core features of Marpuch.

Some other features of Marpuch are:

- Ajax-based keyword suggestion
- Experts will be able to add 'tags'
- Registered users may add some experts as favorite experts
- 'Online Topics' will be shown on main page
- If Marpuch matches experts' country with user’s country, then it will be pointed out. Matching will be done via IP address.
1.3 DESIGN OBJECTIVES

1.3.1 Usability

Marpuch will be an easy to use Web 2.0 application. This is an important requirement for us because our main claim is making life easier for people who search information. In each step we will clearly show all choices of user in one page. This will be done by using Ajax's menu hiding tools.

1.3.2 Reliability

We will keep profiles of registered users and experts. Expert profiles are extremely important because without experts Web 2.0 applications are quite weak. Another important issue is chatting. Chat process should be consistent. Dead links are quite annoying for users so we should not have dead links in our system.

Marpuch may be displayed by different browsers. We should tune Marpuch to as much browser as possible.

1.3.3 Security

Many websites abuse registration process by selling e-mail address of their users or by not taking sufficient security means. Marpuch’s security should be tight. We wouldn’t want to lose our precious volunteers by letting their e-mail address get stolen.

1.3.4 Portability

Marpuch is a Web 2.0 application so it can be accessed from any platform. We will also tune Marpuch to make it nice looking in every browser.

1.3.5 Performance

Performance is an important issue for Marpuch because our aim is to make it faster to get information. Our database should be big but our serve time should also be fast. We use Ajax so Marpuch will not be reloaded every time you make some request.
2 DESIGN CONSTRAINTS

2.1 SYSTEM CONSTRAINTS

2.1.1 HARDWARE CONSTRAINTS

Our hardware requirements are in 3 groups. First one is our hardware requirements for developing process; second one is hardware requirements for server side, last one is requirements for client side.

These are the hardware requirements for developing process:

- Intel Pentium4 3.2 GHz processor
- 1 GB Ram
- 5 GB HDD Space
- Internet connection

These are hardware requirements for server side:

- Intel Pentium Core 2 Duo 2.24 GHz processor
- 2 GB Ram
- 40 GB HDD Space
- Fast Internet connection

These are hardware requirements for client side (non mobile):

- Celeron 1.6 GHz processor
- 256 MB Ram
- Internet connection

These are hardware requirements for client side (mobile):

- Any mobile device which can run jabber client
- Internet connection
2.1.2 SOFTWARE CONSTRAINTS

Our software requirements are in 3 groups. First one is our software requirements for developing process; second one is software requirements for server side, last one is software requirements client side.

These are the software requirements for developing process:

- Windows XP / Linux
- Apache & Tomcat
- NetBeans 5.5 or higher
- JRE
- Smartdraw
- MySQL 5.0
- Gimp
- OpenOffice
- Firefox & IE(6.0,7.0) & Opera & Safari
- JDK
- Microsoft Project

These are the software requirements for server side:

- Apache & Tomcat
- Openfire
- JRE
- Linux

These are the software requirements for client side:

- Any browser
2.2 TIME CONSTRAINTS

Ceng 490 Course syllabus determines our project schedule. Assuming that we will make our final demo on May 2008, we have 6 months left. According to our estimations all team members will have a high workload; however we believe that we can do it just in time if we stick to the Gantt chart.

2.3 PERFORMANCE CONSTRAINTS

We believe that two main topics determine performance constraints. First topic is page loading time; second topic is database query time. AJAX technology is heavily used in Marpuch. This results in less bandwidth usage for Marpuch and less loading time for users. On the other hand we believe that our database design is quite effective so we believe that Marpuch will exceed expectations in performance constraints.
3 ARCHITECTURAL AND COMPONENT LEVEL DESIGN

3.1 MODULES

1. Display Module

This module consists of 3 classes. These classes are:

• Decoder
• GUI
• Displayer

Displayer class is responsible for creating the html codes. It is fed by Decoder and GUI classes. Decoder class sends data about online experts and old answers for that question (or answers of similar question). GUI class sends data about user preferences such as font size, type and so on...

2. Analyzer Module

This module consists of 4 classes. These classes are:

• Parser
• Data Controller
• Special Encoder
• Trigger

Trigger class is responsible for triggering chat between user and expert. It takes user id from Authentication module. It triggers Data Controller class for initiating chat with an expert. Data Controller unit is responsible for jabber related functions. Data controller unit sends chat request to expert and if requested confirmed then it becomes a bridge for chat data between user and expert. Parser unit takes chat data from Data Controller class. If data contains any special tag such as question-answer pair or extension to expert tags then it sends parsed data to special encoder unit otherwise it sends chat data to user. Special Encoder unit is responsible for sending special data such as question-answer pair or expert tags to machine learning module.

3. Machine Learning module

This module consists of 4 classes. These classes are:

• Parser
• Analyzer
• Encoder
• Recorder
Parser class is responsible for parsing data. Analyzer is responsible for creating database queries about searched data. If Analyzer class finds a question matching with searched data or finds online experts about searched subject then it sends those data to Designer module’s decoder class. Analyzer class is fed by Parser class with parsed search data. Encoder class is responsible for encoding question-answer pairs in XML. Encoder class feds Recorder unit with these encoded question-answer pairs. Recorder class is responsible for transforming encoded question-answer pairs into database queries and executing these queries.

4. Authentication Module
This module consists of 4 classes. These classes are:
- Login
- Encryption
- Register
- Authentication

Login class is responsible for getting username-password data, parsing this data and feeding Encryption and Authentication classes with these data. Encryption class is responsible for encrypting password with md5. Encrypted data is later send to Authentication class. Meanwhile username is send to Authentication class by Login class without encryption. Authentication class creates database queries about username-password pairs. If such a pair exists then it sends user id to displayer module, else it alerts displayer module that such a user doesn’t exist by sending an invalid is such as -1. Register class is responsible for creating database queries for adding a new username password pair. Register class uses Encryption class for encrypting password.
3.2 DATA FLOW DIAGRAMS

3.2.1 LEVEL0 DFD

[Diagram showing data flow between USER, EXPERT, DATABASE, and various data flow actions such as ChatRequestToExpert, ChatDataToUser, etc.]
3.2.3 LEVEL2 DFD
1.0 AUTHENTICATION
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Password</td>
<td>The data taken from user to approve log-in</td>
</tr>
<tr>
<td>FilteredData</td>
<td>Data is filtered because of preserving abuse</td>
</tr>
<tr>
<td>EncryptedData</td>
<td>Some data (e.g., password) are used after encrypted to preserve security</td>
</tr>
<tr>
<td>FilteredInfo</td>
<td>The data contains user info and filtered by previous units</td>
</tr>
<tr>
<td>FilteredUserInfo</td>
<td>Gathered data from database</td>
</tr>
<tr>
<td>Name</td>
<td>ApprovalUserLogin</td>
</tr>
<tr>
<td>--------------------</td>
<td>--------------------------------------------------------</td>
</tr>
<tr>
<td>From</td>
<td>AUTHENTICATION UNIT 1.4</td>
</tr>
<tr>
<td>To</td>
<td>GUI UNIT 4.2</td>
</tr>
<tr>
<td>Description</td>
<td>Includes required info about user who has logged-in</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>ApprovalOfUser</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>AUTHENTICATION UNIT 1.4</td>
</tr>
<tr>
<td>To</td>
<td>TRIGGER UNIT 5.3</td>
</tr>
<tr>
<td>Description</td>
<td>Includes required info about user who has logged-in</td>
</tr>
</tbody>
</table>
2.0 INPUT CONTROLLER
### MouseAndKeyboardInput

<table>
<thead>
<tr>
<th>Name</th>
<th>MouseAndKeyboardInput</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>USER</td>
</tr>
<tr>
<td>To</td>
<td>MOUSE EVENT HANDLER 2.2</td>
</tr>
<tr>
<td>Description</td>
<td>Total input from user</td>
</tr>
</tbody>
</table>

### MouseInput

<table>
<thead>
<tr>
<th>Name</th>
<th>MouseInput</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>MOUSE KEYBOARD SPLITTER UNIT 2.1</td>
</tr>
<tr>
<td>To</td>
<td>MOUSE EVENT HANDLER 2.2</td>
</tr>
<tr>
<td>Description</td>
<td>Splitted mouse input by splitter unit. This data will be splitted again for specific uses.</td>
</tr>
</tbody>
</table>

### KeyboardInput

<table>
<thead>
<tr>
<th>Name</th>
<th>KeyboardInput</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>MOUSE KEYBOARD SPLITTER UNIT 2.1</td>
</tr>
<tr>
<td>To</td>
<td>KEYBOARD EVENT HANDLER 2.4</td>
</tr>
<tr>
<td>Description</td>
<td>Splitted keyboard input by splitter unit. This data will be splitted again for specific uses.</td>
</tr>
</tbody>
</table>

### KeyboardInfo

<table>
<thead>
<tr>
<th>Name</th>
<th>KeyboardInfo</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>MOUSE EVENT HANDLER 2.2</td>
</tr>
<tr>
<td>To</td>
<td>TRIGGER UNIT 2.3</td>
</tr>
<tr>
<td>Description</td>
<td>Data that processed by trigger unit to make triggers.</td>
</tr>
</tbody>
</table>

### KeyboardInfoRequest

<table>
<thead>
<tr>
<th>Name</th>
<th>KeyboardInfoRequest</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>TRIGGER UNIT 2.3</td>
</tr>
<tr>
<td>To</td>
<td>KEYBOARD EVENT HANDLER 2.4</td>
</tr>
<tr>
<td>Description</td>
<td>Data that triggers KEYBOARD EVENT HANDLER and the unit gives true data with the help of this data</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ChatRequestFrom</td>
<td>Triggers data control unit to send chat request and related info to Jabber server</td>
</tr>
<tr>
<td>MOUSE EVENT HANDLER 2.2</td>
<td></td>
</tr>
<tr>
<td>TRIGGER UNIT 5.3</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ChatData</td>
<td>Includes raw chat data</td>
</tr>
<tr>
<td>KEYBOARD EVENT HANDLER 2.4</td>
<td></td>
</tr>
<tr>
<td>PARSER UNIT 5.1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SearchData</td>
<td>Raw search data which submitted by user</td>
</tr>
<tr>
<td>KEYBOARD EVENT HANDLER 2.4</td>
<td></td>
</tr>
<tr>
<td>PARSER UNIT 3.1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UsernamePassword</td>
<td>Includes username and password</td>
</tr>
<tr>
<td>KEYBOARD EVENT HANDLER 2.4</td>
<td></td>
</tr>
<tr>
<td>LOGIN 1.1</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>PersonalData</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>From</td>
<td>KEYBOARD EVENT HANDLER 2.4</td>
</tr>
<tr>
<td>To</td>
<td>REGISTER UNIT 1.3</td>
</tr>
<tr>
<td>Description</td>
<td>Includes data that taken from register</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>RateOfExpert</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>MOUSE EVENT HANDLER 2.2</td>
</tr>
<tr>
<td>To</td>
<td>DATABASE</td>
</tr>
<tr>
<td>Description</td>
<td>Rate data includes expert info and rating</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ParsedData</td>
<td>Users search data parsed and passed to analyzer to reach result.</td>
</tr>
<tr>
<td>ErrorFreeExtensibleInfo</td>
<td>Data that will add to database to extent our database and improve our answer ability.</td>
</tr>
<tr>
<td>EncodedDataToMachineLearn</td>
<td>Data is encoded to another form that our application can use.</td>
</tr>
<tr>
<td>OldAnswerAndOnlineExpert</td>
<td>Result of queries to display search results.</td>
</tr>
<tr>
<td>AddNewQuestionAnswer</td>
<td>Data which Filtered and encoded to application form, that submitted by expert. This data is for improving best results capability.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>RecorderQuery</td>
<td>Query from recorder these are mostly insert queries</td>
</tr>
<tr>
<td>File: RECORDER UNIT 3.4</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AnalyzerQuery</td>
<td>Query from Analyzer, these are mostly select queries</td>
</tr>
<tr>
<td>File: ANALYZER UNIT 3.3</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataToDisplay</td>
<td>Includes all online experts data and old question&amp;answers</td>
</tr>
<tr>
<td>File: ANALYZER UNIT 3.3</td>
<td></td>
</tr>
</tbody>
</table>
4.0 DISPLAYER UNIT
<table>
<thead>
<tr>
<th>Name</th>
<th>OnlineExperts</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>DECODER UNIT 4.1</td>
</tr>
<tr>
<td>To</td>
<td>DISPLAYER UNIT 4.3</td>
</tr>
<tr>
<td>Description</td>
<td>Includes online expert information that will be displayed to user</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>OldAnswers</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>DECODER UNIT 4.1</td>
</tr>
<tr>
<td>To</td>
<td>DISPLAYER UNIT 4.3</td>
</tr>
<tr>
<td>Description</td>
<td>Includes old answers data that is relevant to current question</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Styles</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>GUI UNIT 4.2</td>
</tr>
<tr>
<td>To</td>
<td>DISPLAYER UNIT 4.3</td>
</tr>
<tr>
<td>Description</td>
<td>Includes info about registered users’ visual choices such as font, colours ...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Display</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>DISPLAYER UNIT 4.3</td>
</tr>
<tr>
<td>To</td>
<td>USER</td>
</tr>
<tr>
<td>Description</td>
<td>Includes all the data about page that will be shown to user</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>LoginUserInfo</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>GUI UNIT 4.2</td>
</tr>
<tr>
<td>To</td>
<td>DISPLAYER UNIT 4.3</td>
</tr>
<tr>
<td>Description</td>
<td>Includes required info about user who has logged-in</td>
</tr>
</tbody>
</table>
5.0 DISPLAY UNIT
<table>
<thead>
<tr>
<th>Name</th>
<th>QuestionAndAnswer</th>
<th>From</th>
<th>PARSER UNIT 5.1</th>
<th>To</th>
<th>SPECIAL ENCODER UNIT 5.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Data that will be added to database to enrich future results. Contains added question and answer</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>ExtendTag</th>
<th>From</th>
<th>PARSER UNIT 5.1</th>
<th>To</th>
<th>SPECIAL ENCODER UNIT 5.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Data will be added to experts’ tags space. Include tags and expertise levels of expert</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>ChatDataFrom</th>
<th>From</th>
<th>PARSER UNIT 5.1</th>
<th>To</th>
<th>DATA CONTROLLER UNIT 5.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Data that will be sent to Jabber server. Cleared from special tags</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>ChatDataTo</th>
<th>From</th>
<th>PARSER UNIT 5.1</th>
<th>To</th>
<th>DATA CONTROLLER UNIT 5.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Chat data that will parsed by parser unit</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>ChatDataToServer</th>
<th>From</th>
<th>DATA CONTROLLER UNIT 5.4</th>
<th>To</th>
<th>JABBER SERVER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Communication Data</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------------------------------------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ChatDataFromServer</td>
<td>Communication Data</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JABBER SERVER</td>
<td>From JABBER SERVER</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATA CONTROLLER UNIT 5.4</td>
<td>To DATA CONTROLLER UNIT 5.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ChatRequest</td>
<td>Chat request to Jabber Server. Handled by Jabber Server</td>
</tr>
<tr>
<td>DATA CONTROLLER UNIT 5.4</td>
<td>From DATA CONTROLLER UNIT 5.4</td>
</tr>
<tr>
<td>JABBER SERVER</td>
<td>To JABBER SERVER</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ChatDataToUser</td>
<td>Final Data that will be displayed to User</td>
</tr>
<tr>
<td>PARSER UNIT 5.1</td>
<td>From PARSER UNIT 5.1</td>
</tr>
<tr>
<td>USER</td>
<td>To USER</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TriggerData</td>
<td>Trigger Data Controller to</td>
</tr>
<tr>
<td>TRIGGER UNIT 5.3</td>
<td>From TRIGGER UNIT 5.3</td>
</tr>
<tr>
<td>DATA CONTROLLER UNIT 5.4</td>
<td>To DATA CONTROLLER UNIT 5.4</td>
</tr>
<tr>
<td></td>
<td>Description Trigger Data Controller to</td>
</tr>
<tr>
<td>Name</td>
<td>SpecialData</td>
</tr>
<tr>
<td>----------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>From</td>
<td>SPECIAL ENCODER UNIT 5.2</td>
</tr>
<tr>
<td>To</td>
<td>PARSER UNIT 3.1</td>
</tr>
<tr>
<td>Description</td>
<td>Includes data which will extend our database and enrich search results</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>ExtensibleData</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>SPECIAL ENCODER UNIT 5.2</td>
</tr>
<tr>
<td>To</td>
<td>PARSER UNIT 3.1</td>
</tr>
<tr>
<td>Description</td>
<td>Data that will be added to database to enrich our database. These data are given by experts</td>
</tr>
</tbody>
</table>
3.2 STATE TRANSITION DIAGRAM
4 OBJECT-ORIENTED DIAGRAMS

4.1 USE CASE

- Registering
- Logging In
- Chatting
- Asking Question
- Rating Experts
- Adding Expertise
- Adding Question-Answer Pair
- Reporting Abuse
- Reading Help
- Pinning Question
- Adding an Expert to Favorite Experts
4.2 ACTIVITY DIAGRAMS

4.2.1 REGISTERING

When one of our clients (users) comes to our Marpuch server he/she may want probably to do will be registering if he/she will not registered already. Although registration is not mandatory to ask questions, it is a must for pinning questions, rating experts and adding experts to ‘favorite experts’ list. Experts need to register as well. Users need to fill these fields:

- **Username**: Nick name of user. This is a mandatory field. We will control uniqueness of lowercase of nickname, which is username can include uppercase characters but our system is not case sensitive when processing nicknames. A validator controller will check this box and warn if user chooses a nickname which is already registered or nickname textbox is empty.
- **Password**: Password of user. This is a mandatory field. This field is disabled until a unique nickname is entered. Passwords will be shown as stars. It will be encoded when inserted to database. Password must be at least 6 character and at most 12 characters. Moreover it must contain a numeric and alpha numeric character.
- **Name**: Real name of user. This is another mandatory field. It is also disabled until password and username are entered correctly.
• Surname: Real surname of user. This is another mandatory field. It is also disabled until password, username and Name fields are entered correctly.
• Location: Which country is user from? It consists of 3 different parts.
  o Country: This is a dropdown menu which contains all the countries of the world.
  o City: This is also a dropdown menu which is filled after the user selects his country.
  o Zip code: This textbox is enabled when user chooses city part. This part is not mandatory
• E-mail address: This textbox is enabled after mandatory fields in previous part are filled correctly. This is a mandatory field. We will use this e-mail address when a user’s pinned question is answered and also more importantly it will send a new password and his old username (nickname) to users and experts who forget his/her password. Validater checker will check whether e-mail address is syntaxly valid or not.
• Education Level: This is a mandatory field. It is for users and experts to know each other’s knowledge level. It is enabled after mandatory fields in previous part are filled correctly.
• Department: This is implemented with Ajax hiding facility. So it is visible only if education level is either under graduation or graduation.
• Become an Expert: This is not a mandatory field. If the users check “I want to be an expert” checkbox then hidden expertise subjects (tags) will be shown in a table in alphabetical order with help of Ajax. Expert candidate can select tags from this table with the help of check boxes.
• Instant Messaging Address: This part is enabled after previous mandatory parts are filled correctly and also “Became an Expert” checkbox is checked. This is a row which consists of three basic html elements. These elements are textbox, label, and dropdown list. Account name is written in textbox, “@” is written in the label. Available instant messaging servers will be in the dropdown menu. User will chose one of them. We cannot allow every instant messaging server because some instant messaging servers do not support Jabber. Marpuch’s jabber server will make a friend request to this address. If the volunteer accepts Marpuch server’s friend request, Marpuch will notify him/her that he/she become an expert. From now on expert will use this IM address whenever he/she chats with users.
• Register Button: This button will be enabled only if the user fills all steps correctly. When this button is clicked user will be registered to Marpuch’s database and he/she will be able to login. Moreover he/she will be redirected to Marpuch’s main page as a logged in user.
4.2.2 LOGIN

User enters his/her email and clicked the button

User requests to login

User is redirected to index page

User enters user name and password

User name and password

User clicks login button

User login button

User logged in

User successful login

New password and old password have been sent to users' email address

User prompts to login

User is redirected to index page

User is redirected to reset password

User is redirected to login

User is redirected to main page
Registered users may want to probably log in. Although login is not a mandatory process for asking questions to experts, when a user logs in he/she will be able to look for their favorite experts. Moreover users must login in order to pin unanswered questions. Experts do not need to login to answer questions since question-answering process is done via instant messaging application. Users, experts and moderating stuff use same page for logging in.

Users/Experts need to fill these fields:

- **Username:** Username comparisons are case-insensitive. Validate checker warns if username doesn’t exist.
- **Password:** Password comparisons are case-sensitive. Validate checker warns user if password is less than 6 characters or more than 12 characters.
- **Login button:** When a user fills both of these fields, this button will be enabled. Login process works this way: Username will be transformed to lowercase. Password will be encrypted. This translated username and encrypted password will be compared with database usernames and passwords which are in the entity named “Membership”. If the login process success user will be redirected to main Marpuch page. Otherwise user will be redirected to login page.
- **Forget Password?** : This is an Ajax based button. When this button is clicked, a hidden textbox and a button will appear. When the user enters his/her email address and click the button, Marpuch’s database will make a control if the email address entered by the user exists. If email address exists then a new password and old username will be sent to this email address. However if the email address does not exist an Ajax based notification will be shown with the text “There is no match with this email address”.


4.2.3 ASKING QUESTION

Another user demand can be asking a question to an expert. For this purpose user must redirected to ask question page. This page is where users enter their question. Marpuch parses this question and search its database for an appropriate answer and similar questions in that subject.

Users need to fill these fields:
• Question Box: Users enter their questions in English language to this Ajax based textbox. Questions should be meaningful English sentences so that Marpuch can extract as much detail as possible with the help of Marpuch’s machine learning tool. If user writes only keywords such as “Firefox, c++, Athletics etc.” Marpuch will still be able to offer possible questions-answers and experts. Marpuch will be able to suggest question completion with the help of Ajax.

Ask Button: This button will trigger the Marpuch’s machine learning tool to parse the question, also trigger the Marpuch’s database for searching online experts which are online and relevant to question’s subject. Moreover user will be redirected to the question-answering page.
4.2.4 QUESTION ANSWERING

When a user asks a question he/she want to get an answer to his/her question. In this situation user will be redirected to question answering page. This page has mainly three parts. First part will contain answer of the question if Marpuch’s machine learning tool finds an appropriate answer from its database. Second part will consist of the online expert list. There will be boxes in this list. Each box will contain nickname of expert, chat count of expert, education level of expert and also some extra information about expert. Third part is a list of relevant questions on that subject.

Users need to fill these fields:
• Chat buttons: Each of these buttons is inside experts’ information boxes. When this button is clicked a chat request will be send to the selected expert’s instant messaging application via Marpuch’s jabber server. If the expert confirms chat request then Ajax based chat interface will be opened.

• Relevant questions: This is an Ajax based hyperlink. When it is clicked a hidden form will be shown. In this form Marpuch will show question and answer pairs.
4.2.5 CHATTING

User prompts to chat with an expert → Sent a request to expert for starting chat

[cancel]

branch

[accept]

Chat interface has been opened

Chat started

merge

Ask a question

else

Send a question

Get an answer

branch

[finish]

Rate expert
After selection of an expert in “Question Answering” page chat will be started between expert and user. User ask his questions to expert from an interface, this is an Ajax based interface looks like msn chat interface, which contains chat history, send button, rating box and a textbox for writing a message.

- Textbox: User and expert will write their messages here. Expert can enter specific key words here. These key words are:
  - <tag="expertise", level="number (between 1 to 10)"/>
  - <question="question sentence", answer="answer sentence"/>

- Rating box: There are five stars here. Experts’ average rating will be shown by coloring these starts in red. User will be able to rate experts by clicking these stars. An example of this rating system can be seen at “www.youtube.com”.

• Textbox: User and expert will write their messages here. Expert can enter specific key words here. These key words are :
  o  <tag= “expertise”, level= “number (between 1 to 10)” /> 
  o  <question = “question sentence”, answer = “answer sentence”/>

• Rating box: There are five stars here. Experts’ average rating will be shown by coloring these starts in red. User will be able to rate experts by clicking these stars. An example of this rating system can be seen at “www.youtube.com”.


4.2.6 READING HELP

- User made a request for help
- User selected a help topic
- User is redirected to help page
- User is redirected to "How to become an expert?"
- User is redirected to "Getting started page"
- User is redirected to "How to become an expert from the community?"
- User is redirected to "My account and policies"
- User is redirected to "Troubleshooting page"
- User is redirected to "Get help and policies"
User can take help by reading Marpuch’s help page. There are 6 main help topics in this page. These main topics are:

1. Getting Started: This part includes a glossary and information about many subjects such as: account types, how a question should be asked to get a better result.

2. Becoming an Expert: This part explains how to become an expert in details.

3. The Marpuch Community: This part is all about Marpuch community. This part includes information about:
   a. How to add an expert to favorite experts list.
   b. What does it means adding an expert to favorite experts list?
   c. How do I invite my friends to join Marpuch?

4. Account and Policies: This part includes information about:
   a. I forget my username and password. How do I log back to Marpuch?
   b. How do I close my account?
   c. How do I report abuse?
   d. How do I rate experts?
   e. Why was my account suspended?
   f. How do I block another user?
   g. How do I change my password?

5. Advanced: This part includes information about:
   a. Will my mobile device work with Marpuch Mobile?
   b. How do I add a new expertise subject?
   c. I am expert an area which is not in suggested expertise list. What do I do?
   d. I want to add a new question-answer pair which deducted from a chat with a user. What do I do?

6. Troubleshooting: This part includes information about:
   a. Solving General Site Issues
   b. Common Error Messages
4.2.7 PINNING A QUESTION

Users can pin a question if Marpuch cannot find an appropriate answer and all expert in that area are offline. It is shown as a suggestion in question-answering page.
4.2.8 REPORTING ABUSE

Users and experts can report abuse by clicking abuse link. Member will write abusive users’ name. Then he/she will describe the abuse type.
4.2.9 REPORTING BUG

Users and experts can report bugs by entering report bug page. Then he/she will select his/her operating system from a dropdown box, his/her browser from dropdown box, the page he/she encountered the bug from also a dropdown box. After that he/she writes a short description of bug into a textbox.
4.3  CLASS DIAGRAMS

4.3.1 AUTHENTICATION MODULE

```
Login
-UserName : string
-Password : string
+setNamePassword(UsernamePassword : string) : void
+parseData() : void
+getPassword() : string
+getUsername() : string

Authentication
-EncryptedPassword : string
-Username : string
-Query : string
-ValidateUserId : int
+setName(Username : string) : void
+setEncryptedPassword(Password : string) : void
+getQuery() : void
+validateLogin() : int

Encryption
-Key : string
-Pass : string
+setData(Data : string) : void
+encryptData(Data : string) : string
+decryptData(Data : string) : string

Register
-PersonalData : string
-Query : string
+setPersonalData(PersonalData : string) : void
+filterData(PersonalData : string) : string
+parseData() : void
+prepareQuery() : void
+insertionQuery() : void
```
4.3.2 MACHINE LEARNING MODULE

```
Parser
- SearchData : string
- extensibleData : string
- parsedSearchData : string
- parsedExtensibleData : string
+ setSearchData(data: string) : void
+ setExtensibleData(data: string)
+ parseSearchData(searchData: string) : string
+ parseExtensibleData(extensibleData: string) : string
+ getparsedSearchData(void) : string
+ getparsedExtensibleData(void) : string

Analyzer
- Query : string
- ResultOfQuery : List
- preparedData : string
+ setQuery(query: string) : void
+ sendQuery(void) : void
+ prepareDataToDisplay(void) : void
+ getPreparedData(void) : string

Encoder
- ErrorFreeExtensibleInfo : string
- EncodedDataToMachineLearning : string
+ setErrorFreeExtensibleInfo(data: string) : void
+ getEncodedDataToMachineLearning(void) : string
+ encode(void) : void

RecorderUnit
- Query : string
- rawData : string
+ setRawData(rawData: string) : void
+ createQuery(void) : void
+ sendQuery(void) : List
```

List
4.3.3 DISPLAY MODULE

![Class Diagram]

- Decoder
  - DataToDisplay : string
  - OnlineExperts : string
  - OldAnswers : string
  + setDataToDisplay(data: string) : void
  + parse() : void
  + getOnlineExperts() : void
  + getOlAnswers() : void

- Displayer
  - Styles : string
  - RegisteredUserInfo : string
  - OnlineExperts : string
  - OldAnswers : string
  + setOnlineExperts(onlineExpert: string) : void
  + setOldAnswers(oldAnswer: string) : void
  + setStyles(style: string) : void
  + setRegisteredUserInfo(userInfo: string) : void
  + createHtmlPage() : void

- GUI
  - ApprovalUserLogin : int
  - getStyleQuery : string
  - resultOfQuery : List
  - styles : string
  + setApprovalUserLogin(userId: int) : void
  + prepareQuery() : void
  + sendQuery() : List
  + encodeStyles() : void
  + getStyles() : string
4.3.3 ANALYZER MODULE

Parser
- chatData : string
- chatDataToUser : string
- questionAndAnswer : string
- extendTag : string
- chatDataFrom : string
- chatDataTo : string

+ setChatData(data : string) : void
+ getChatDataToUser(void) : string
+ getQuestionAndAnswer(void) : string
+ getExtendTag(void) : string
+ getChatDataFrom(void) : string
+ set(chatDataTo(data : string)) : void
+ parseChatDataToAndGetExtensibleInfo(void) : void

DataController
- chatDataFrom : string
- chatDataTo : string
- triggerData : bool
- approvalOfExpertChat : bool
- chatDataToServer : string
- chatDataFromServer : string

+ setChatDataFrom(data : string) : void
+ getChatDataTo(void) : string
+ setTriggerDataTo(TriggerData : bool) : void
+ requestConnection(void) : bool
+ sendChatDataToServer(chatDataToServer : string) : void
+ setChatDataFromServer(void) : string

SpecialEncoder
- QuestionAndAnswer : string
- ExtendTag : string
- SpecialData : string
- ExtensibleData : string

+ setQuestionAndAnswer(QuestionAndAnswer : string) : void
+ setExtendTag(ExtendTag : string) : void
+ getSpecialData(void) : string
+ getExtensibleData(void) : string
+ encodeExtensibleData(void) : void

Trigger
- approvalOfUser : int
- chatRequestFrom : bool
- triggerData : bool

+ setApprovalOfUser(userId : int) : void
+ setChatRequestFrom(req : bool) : void
+ setTriggerData(val) : void
4.4 SEQUENCE DIAGRAMS

4.4.1 AUTHENTICATION MODULE
4.4.2 MACHINE LEARNING MODULE
4.4.3 DISPLAYER MODULE
4.4.4 ANALYZER MODULE
5 USER INTERFACE DESIGN

5.1 INTRODUCTION

Marpuch is a system that connects people to people using web. In a web application user interface is too important. Interface must be clear and easy to use. The user can reach every feature easily.

Below we will explain main points of graphical user interface.

5.2 HOME PAGE
This is the main page of Marpuch. Main page has a Google-like interface. It is as simple as possible. In home page there is a textbox that can be used to enter keywords. There are also links to other pages which are include other features of “Marpuch”. These links are for:

- Be an expert
- Help
- Report Abuse
- Report Bug
- About Marpuch
- About Us

These features will be explained below.

This interface will be improved and modified later. But main idea is same “Simple Design, Intense Content”.
5.3 SEARCH

User enters keyword(s) to the textbox on home page, and hits “Find An Expert” button. Marpuch engine recognizes keywords and prepare results that contains experts that are related on these keywords.

In results page there are boxes which includes short information about experts. These boxes are also links to the experts’ profile pages. User looks to experts and choses one of them and marpuch redirects user to expert’s profile page.

In Marpuch’s final version this page also includes old answers that are given to related questions. Since these screenshots are taken from pre-prototype version of Marpuch many features are missing and will be completed in final release.

This page contains more information for registered users such as offline experts, prefered experts etc...

This page also includes a textbox that works same as textbox which is in home page. User can make a new search or make specific last search to get better results.
5.4 START CHAT

This page is a sample expert profile page. In this page there will be all information that will displayed to user. User looks experts’ information and decides whether start chat or not. If user decided to start chat clicks “Start Chat” button and Marpuch send request to expert to start chat.
5.5 CHATTING

Above is the chatting screen. In the middle of the main page there is a pop-up window. This window has msn-like structure and it has a simple IM function.

User uses this window to communicate with expert but expert uses own instant messaging application. Expert must use Jabber-based messenger account. Details of chatting explained on 4.2.5.
5.6 REGISTRATION

Above is the registration page. Since user presses “Be An Expert” button, a modal pop up window. This window is AJAX-based and MacOS style.

When user is filling areas, an AJAX-based controls the inputs and give feedback to user immediately. For example when user is filling mail adress, marpuch invokes a javascript function and function controls the input using regular expressions.

After filling all the areas, user clicks “Create My Account” button. If the process is finished succesfully, the user info inserted to system.
5.7 HELP

Above is the help page. Help topics are grouped. These headlines are links to specific pages. In help page, there are useful information for users and experts. Also there are solutions to common problems.
5.8 REPORT BUG

Above is the “Report Bug” page. User or experts use this page to submit bugs that they have faced. Reporting bug is one of the main key points of improving our application. To specify the problem we have prepared some predefined choices. User selects suitable choice for them and give a brief description of bug. Our predefined choices are

- Type Of Problem
- Location Of Problem
- Operating System
- Browser
5.9 REPORT ABUSE

Above is the “Report Abuse” page. Since every user can be an expert, there may be some non-ethical problems. System can control some of the words but it is impossible to control whole conversation.

User can be complaint about experts or vice versa. If an expert/user wants to complaint another user/expert, fills this form and submit. These forms will be processed by our system regularly.
5.10 ABOUT MARPUCH

Above is the “About Marpuch” page. This page contains brief description of system and Marpuch’s advantages due to other question-answer systems. Also there are some information about AJAX.

Aim of this page is give people an introduction about Marpuch.
5.11 ABOUT US

Above is the “About Us” page. There is a picture of founders of Company and contact addresses.
6 PROCESS

6.1 TEAM STRUCTURE

We have decided that Controlled Decentralized (CD) team structure fits us. Our team has a leader and our leader assigns sub topics to members. We have a collaborative decision mechanism. Communication between members is really important. We have meetings in strict hours and every time we are in touch.

6.2 PROCESS MODEL

We will follow a step by step program. We will do analysis, initial design, more detailed design, prototype, implementation, testing & debugging and maintenance. The most suitable process model is “Waterfall Model” for us. After we finished one step we won’t turn back to previous steps. So, all the features of a step must be well-understood and applied.

6.3 GANTT CHART

Gantt chart is on Appendix section.
7 TESTING

7.3 INTRODUCTION

In order to present a robust, bug-proof product we will make many tests on Marpuch. Since we have a strict time table, testing will coincide with developing. We will divide testing into 4 parts:

Unit Testing → Integration Testing → Validation Testing

7.4 TESTING STRATEGY

We will make unit testing while creating modules. After implementing a class we will test that unit. If we find any bug then we will focus on fixing that bug. We will assume that a module takes correct input from another module; that is we will not try to fix inter-module bugs in unit testing. Inter-module bugs will be fixed during integration test. In validation testing, we will check whether our product runs on different platforms or not. During unit testing and integration testing, we will work on Windows XP© operating system with Mozilla Firefox© browser.

7.5 UNIT TESTING

- Displayer Module

<table>
<thead>
<tr>
<th>Name: Are all online experts listed?</th>
</tr>
</thead>
<tbody>
<tr>
<td>How will we do the test? We will inspect the page with 5-40 experts.</td>
</tr>
<tr>
<td>If Succeeds: Pass to next unit test.</td>
</tr>
<tr>
<td>If Fails: Focus on Decoder and Displayer classes to find the bug.</td>
</tr>
<tr>
<td>Name:</td>
</tr>
<tr>
<td>-------------------------------</td>
</tr>
<tr>
<td>How will we do the test?</td>
</tr>
<tr>
<td>If Succeeds:</td>
</tr>
<tr>
<td>If Fails:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name:</th>
<th>Is user info shown correctly?</th>
</tr>
</thead>
<tbody>
<tr>
<td>How will we do the test?</td>
<td>We will inspect the page with different users including anonymous user</td>
</tr>
<tr>
<td>If Succeeds:</td>
<td>Pass to next unit test.</td>
</tr>
<tr>
<td>If Fails:</td>
<td>Focus on Displayer and GUI classes to find the bug.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name:</th>
<th>Is user’s preferred font type used in page?</th>
</tr>
</thead>
<tbody>
<tr>
<td>How will we do the test?</td>
<td>We will inspect the page with different users who have distinct font type preferences.</td>
</tr>
<tr>
<td>If Succeeds:</td>
<td>Pass to next unit test.</td>
</tr>
<tr>
<td>If Fails:</td>
<td>Focus on Displayer and GUI classes to find the bug.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name:</th>
<th>Is user’s preferred font size used in page?</th>
</tr>
</thead>
<tbody>
<tr>
<td>How will we do the test?</td>
<td>We will inspect the page with different users who have distinct font size preferences.</td>
</tr>
<tr>
<td>If Succeeds:</td>
<td>Pass to next unit test.</td>
</tr>
<tr>
<td>If Fails:</td>
<td>Focus on Displayer and GUI classes to find the bug.</td>
</tr>
</tbody>
</table>
Name: Is user’s preferred page layout used in page?

How will we do the test? We will inspect the page with different users who have distinct page layout preferences.

If Succeeds: Pass to next unit test.

If Fails: Focus on Displayer and GUI classes to find the bug.

• Authentication Module

Name: Is usernamePassword parsed correctly?

How will we do the test? We will test by using 20 distinct username-password pairs.

If Succeeds: Pass to next unit test.

If Fails: Focus on Login class to find the bug.

Name: Are passwords encrypted in md5 correctly?

How will we do the test? We will compare 20 results with 2 free md5 encryption programs results.

If Succeeds: Pass to next unit test.

If Fails: Focus on Encrypt class to find the bug.

Name: Is username-encrypted password pair added to database correctly?

How will we do the test? We will use MySQL command line client to check results.

If Succeeds: Pass to next unit test.

If Fails: Focus on Register class to find the bug.
Name:  Is username-encrypted password pair retrieved correctly?

How will we do the test?  We will use MySql command line client to check results.

If Succeeds:  Pass to next unit test.

If Fails:  Focus on Authentication class to find the bug.

•  Analyzer Module

Name:  Is Data Controller class always triggered?

How will we do the test?  We will continuously do this test, by sending chat requests.

If Succeeds:  Focus on Trigger and Data Controller classes to find the bug.

If Fails:  

Name:  Is Data Controller class sends data to jabber server correctly?

How will we do the test?  We will check send data from expert’s messenger window.

If Succeeds:  Pass to next unit test.

If Fails:  Focus on Data Controller class to find the bug
<table>
<thead>
<tr>
<th>Name: Is Data Controller class takes data from jabber server correctly?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>How will we do the test?</strong></td>
<td>We will send 10 data via expert’s messenger window.</td>
</tr>
<tr>
<td><strong>If Succeeds:</strong></td>
<td>Pass to next unit test.</td>
</tr>
<tr>
<td><strong>If Fails:</strong></td>
<td>Focus on Data Controller class to find the bug</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name: Is data parsed correctly?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>How will we do the test?</strong></td>
<td>We will send 10 messages to check parser.</td>
</tr>
<tr>
<td><strong>If Succeeds:</strong></td>
<td>Pass to next unit test.</td>
</tr>
<tr>
<td><strong>If Fails:</strong></td>
<td>Focus on Parser and Data Controller classes to find the bug.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name: Are special messages from expert send to Special Encoder class correctly?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>How will we do the test?</strong></td>
<td>We will send 20 message, 10 of which are special messages and inspect parser’s reaction.</td>
</tr>
<tr>
<td><strong>If Succeeds:</strong></td>
<td>Pass to next unit test.</td>
</tr>
<tr>
<td><strong>If Fails:</strong></td>
<td>Focus on Parser and Special Encoder classes to find the bug.</td>
</tr>
</tbody>
</table>
**Machine Learning Module**

<table>
<thead>
<tr>
<th>Name:</th>
<th>Is search data parsed correctly?</th>
</tr>
</thead>
<tbody>
<tr>
<td>How will we do the test?</td>
<td>We will send 10 search data to check parser.</td>
</tr>
<tr>
<td>If Succeeds:</td>
<td>Pass to next unit test.</td>
</tr>
<tr>
<td>If Fails:</td>
<td>Focus on Parser class to find the bug.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name:</th>
<th>Is extensible data parsed correctly?</th>
</tr>
</thead>
<tbody>
<tr>
<td>How will we do the test?</td>
<td>We will send 10 search data to check parser.</td>
</tr>
<tr>
<td>If Succeeds:</td>
<td>Pass to next unit test.</td>
</tr>
<tr>
<td>If Fails:</td>
<td>Focus on Parser class to find the bug.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name:</th>
<th>Is parsed extensible data encoded correctly?</th>
</tr>
</thead>
<tbody>
<tr>
<td>How will we do the test?</td>
<td>We will send 10 extensible data to check parser.</td>
</tr>
<tr>
<td>If Succeeds:</td>
<td>Pass to next unit test.</td>
</tr>
<tr>
<td>If Fails:</td>
<td>Focus on Parser and Encoder class to find the bug.</td>
</tr>
</tbody>
</table>
### Initial Design Report | TOMBEKI++

#### Name: Are analyzer queries created correctly?

**How will we do the test?**
We will use MySQL command line client to check results.

**If Succeeds:**
Pass to next unit test.

**If Fails:**
Focus on Analyzer class to find the bug.

#### Name: Are question-answer pairs added to database correctly?

**How will we do the test?**
We will use MySQL command line client to check results.

**If Succeeds:**
Pass to next unit test.

**If Fails:**
Focus on Recorder class to find the bug.

### 7.6 INTEGRATION TESTING

#### Name: Are extensible data sent to Artificial Intelligence module from Analyzer module correctly?

**How will we do the test?**
We will use 10 question-answer pair and 10 tag extension to check for bugs

**If Succeeds:**
Pass to next integration test.

**If Fails:**
Focus on AI:: parser and Analyzer:: specialEncoder classes to find the bug.
<table>
<thead>
<tr>
<th>Name:</th>
<th>Is user id data sent to Analyzer Module from Authentication Module correctly?</th>
</tr>
</thead>
<tbody>
<tr>
<td>How will we do the test?</td>
<td>We will use 10 distinct user accounts and also an anonymous user to check for bugs.</td>
</tr>
<tr>
<td>If Succeeds:</td>
<td>Pass to next integration test.</td>
</tr>
<tr>
<td>If Fails:</td>
<td>Focus on Analyzer:: trigger and Authentication:: authentication classes to find the bug.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name:</th>
<th>Are old answers and online experts’ data sent to Displayer module from Artificial Intelligence module correctly?</th>
</tr>
</thead>
<tbody>
<tr>
<td>How will we do the test?</td>
<td>We will use 5 online experts’ data and 10 old answers to check for bugs</td>
</tr>
<tr>
<td>If Succeeds:</td>
<td>Pass to next integration test.</td>
</tr>
<tr>
<td>If Fails:</td>
<td>Focus on Displayer:: decoder and AI:: analyzerUnit classes to find the bug.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name:</th>
<th>Is user id data sent to Displayer module from Authentication module correctly?</th>
</tr>
</thead>
<tbody>
<tr>
<td>How will we do the test?</td>
<td>We will use 10 distinct user accounts and also an anonymous user to check for bugs.</td>
</tr>
<tr>
<td>If Succeeds:</td>
<td>Pass to next integration test.</td>
</tr>
<tr>
<td>If Fails:</td>
<td>Focus on Displayer:: GUI and Authentication:: authentication classes to find the bug.</td>
</tr>
</tbody>
</table>
## 7.7 VALIDATION TESTING

<table>
<thead>
<tr>
<th>Name:</th>
<th>Is Marpuch working correctly on IE, Safari, Konqueror, Opera, Netscape browsers? (Firefox will have already been tested)</th>
</tr>
</thead>
<tbody>
<tr>
<td>How will we do the test?</td>
<td>We will use each of the browsers</td>
</tr>
<tr>
<td>If Succeeds:</td>
<td>Pass to next validation test.</td>
</tr>
<tr>
<td>If Fails:</td>
<td>Focus on browser specific tags.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name:</th>
<th>Is Marpuch working correctly on Linux? (Microsoft will have already been tested)</th>
</tr>
</thead>
<tbody>
<tr>
<td>How will we do the test?</td>
<td>We will use Marpuch in pardus and ubuntu operating systems.</td>
</tr>
<tr>
<td>If Succeeds:</td>
<td>Pass to next validation test.</td>
</tr>
<tr>
<td>If Fails:</td>
<td></td>
</tr>
</tbody>
</table>
8 CONCLUSION

We benefit much from requirement analysis report so when we started initial design report we were aware of the importance and benefits of initial design report. We are quite aware of the importance of good design so we give our best to design our project and prepare this document. We believe that this report exceeds expectations and will not only help us in the coming months but also will help next generation web service developers in their design phases. We changed some of design choices since requirement analysis report so we believe that writing an initial design report will reflect our project much more better than requirement analysis report.

We were not sure about how testing will be done until we start writing this document. Now we determine not only how we will do the test but also what kind of tests we will do, when will we do tests and how we will do the tests.

Modules were determined in requirement analysis report, we also determined classes of each module with their member functions in initial design report and we believe that it will make our job easy in the coming months.

To summarize we believe that tombeki++ team will get benefits of initial design report in the coming weeks. This report will guide us throughout the term. With the feedbacks from course coordinators we will update our design.
9 REFERENCES

• http://en.wikipedia.org

• Software Engineering A Practitioners Approach. Pressman S. Roger

• Unified Modeling Language User Guide, 2nd