Up to now we resolved some implementation issues of the web application, we successfully connected to web service and called some methods with parameters from plugin, created some database tables, began implementing data access classes.

Erdem KARAHAN
e1347616

Last status of my progress was congested by means of passing parameters by the client side web service applications, by which I was about to use for sending the receiver's identity, sender's identity, the date of the message and the message body, for an arbitrary message I created(not the actual message sent by the IMBO application). Even if I created successful connection, and received acknowledge, it was impossible for me to successfully pass the parameters I mentioned. However, now I'm able to successfully send the parameters thanks to the axis2 wsdl2java wizard. Using the classes generated by the tool, and additional coding that uses the classes generated by the wsdl2java tool, I'm able to pass the necessary parameters that ensure me I will be able to satisfy any communication necessary for future applications. Now I'm looking forward to continue coding the plug-in to extend the work provided up to now.

Taha Bekir EREN

Up to now, I have studied on javascript and AJAX. Since we have a pull architecture in which client javascript pulls data from the server via web methods, we have to use javascript to perform necessary actions on the client side. While learning javascript, I resolved some implementation issues. For now, I'm planning to pull three kind of data from server which are:

1. Login event notification
2. Incoming message
3. Roster change notification
4. Error messages

Since XMPP login is an asynchronous operation, we need a mechanism that
will notify the client when login is completed at the server. To make this notification possible, when client pulls data from the server, a special string is sent to client which means login is successful. After receiving this message, client javascript redirects the user from login page to the main page of our application.

When an incoming message is received at the server, we will save it to a queue which will be specific to each user. When the client pulls data, it will receive a message body and sender information. At this point client has to resolve if there is any active chat window between the user and the sender of the received message. If there is, then the message body will be appended to the related chat window, else a new popup chat window will be created and populated with neccessary data. By this mechanism we centralize the message receiving operation. On the contrary, each chat window will send its own message via a web method call.

Change of roster is somewhat complicated with respect to client side. At first I planned to manipulate the html document by javascript when a roster change happens. But I think it will be too manual and unnecessary work to do so. Instead of this I will create a aspx page for roster, and embed it to our main page via iframes. When the client receives a roster change notification, it will simply reload the related frame.

Another issue is error messages. We have to inform the user that some error has occured, such as an unsuccessful login. When the client receives an error message, it will display an alert box with the error message as text.

---

Mine Karakaya
The Database layer module has gone into revision during the period according to the needs of the project. I implemented the tables and according constrains; in and between tables; in SQL Server. Till this time, I have implemented the functions of the database layer module relating with Users and Users_and_Groups classes completely in ADO.NET from Visual Studio. I am now working on searchGroup function about the Groups table; when it is done Groups table related functions will be implemented completely. As I have implemented the functions; I have created separate forms for each of the tables in Visual Studio in order to achieve unit testing. In the forms buttons named with functions and textboxes related to each function parameters take place.

Next week, we will start implementing User class (web application) and functions related to Group_Pictures, Group_Walls and Profiles tables (dblayer).