Instant Messaging: Architectures and Concepts

Student: Linan Zheng
Mentor: Dr. Stephan Rupp

Institute of Communication Networks and Computer Engineering
University of Stuttgart

Abstract

Instant Messaging (IM) is an Internet-based protocol application that allows one-to-one communication between users employing a variety of devices. [1] Recently, Instant Messaging has already obtained the remarkable success as P2P communication tool. In some places, it already took the place of e-mail as the first choice for long distance communication. In the mobile area, the Short Message Services (SMS) and Multimedia Messaging Service (MMS) also attract thousands of subscribers by the richer and richer set of services. Jabber is the most widespread open source platform, using an XML encoded protocol, especially tailored to provide instant messaging and presence services over the Internet.

This paper gives some basic concepts of IM System and analyzes the basic architecture, protocol, and communication procedure based on Jabber to briefly describe the working flow of typical IM system. At the end, by comparing ICQ, MSN and Yahoo Messager - 3 most popular IM software in the nowadays market, we can get a deeper understanding of their features and working principle.

Keyword Instant Messaging, Jabber, XMPP, presence, ICQ, MSN, Yahoo Messager

1 Introduction

1.1 What is IM system?

Today, Instant Message is widely applied. By this kind of software, people who log online can get response in a short time after sending message which is a way to real-time communication and chatting. For most people, it is cheaper than chatting by telephone; furthermore, the newest IM software integrated data transmission, voice chatting, video conference, e-mail and so on. Some analysts call IM as “real-time e-mail.” For enterprises, IM exploits new area for web-services. People approve IM software not only for fun and chat, but also for the excellent performance in business communication. The attraction in business is obvious to see: for one hand, it will tell the stuff weather other colleagues online, to prevent wasting time on telephone or informing conference and this kind of affairs. On the other hand, it provides a real-time dialog to catch the business opportunity, especially for the interaction between companies and their customers. [2]

As shown in Figure 1, IM is derived from UNIX talk program since around 1970 which only offer pure message sending. People use it only at home or office. Later, more functions were added and people can use it anywhere with PC, laptop, PDA or a mobile phone. Now IM services have got wide world known for allowing people all over the globe to connect in almost real-time with anyone else in the world. In the market, there are ICQ, AOL, MSN, Yahoo Messenger, and a lot of similar applications.
Most of these applications rely on a central server that administers users and informs others about their presence and location, so a user can send directly messages to another user, if its location is already known. See Figure 2.

**1.2 presence awareness**

What is PRESENCE? It is equal to indication of availability of communication. A user, a device or some resource offer some ways to communicate: Do I want to pick up the phone call? Can you talking to your friend by video cam? Where is the college you need to find? Are there any free seats in the meeting room? And so on. The presence information includes means, ability, capability, status, willingness and location. By publishing presence information, it is getting easier to contact people or to utilize resource. Meantime, it helps the current service offering much better service according to customers’ different presence status. As shown in figure 3, presence function integrated user’s location, time, schedule and preference information to give a vivid display of people’s status. The concept of presence has received substantial attention from the virtual reality community, and is becoming increasingly relevant both to broadcasters and display developers.
The IETF published the Presence and Instant Messaging Model, RFC-2778 in Feb.2002 [5]. The model contains the several entities involved, a description of the basic functions they perform, and more relevant, the definition of a common vocabulary that can be used to facilitate discussions among different systems.

Let’s take a look at figure 4, it’s a simplified concept model the presence system collects and publishes presence information to users who interested in this information, including personal, application and services. The policy database controls the communication of the presence information by Buddy List or judging who and when can see the information. The presence service consists of two types of clients: Presentity and Watchers. The Presentity provides presence information to the system which is collected from several Presence User Agent (UA), as PC stared, phone is busy or cell phone without signal. The Watcher can either take the form of a fetcher or a subscriber. The fetcher simply requests the current state of Presentity from the presence service. On the other hand, a subscriber requests notifications from the presence service of future changes of some Presentity. There is a special kind of fetcher called Poller. The Poller ‘fetches’ presence information on regular basis. The changes of presence information are distributed to subscribers via
notifications. The presence service also keeps information about watchers and its activities in terms of fetching or subscribing to presence information. The presence service may also distribute watcher information to other watchers using the same mechanisms that are available for distributing presence information. The model defines the presence protocol as the interaction between presence service, Presentities and Watchers.

2 Analysis of Jabber based IM system

2.1 What is Jabber?

Jabber is a freely available set of streaming XML (Extensible Markup Language) protocols and technologies. The development of this project started in the year 1998 by Jeremie Miller. The first application available on Jabber platform is the instant messaging systems, which focus on privacy, security, ease of use and provides standard IM functionality, such as messaging, contact lists management and distribution of presence information.

Jabber is a server-based program that runs on a system of distributed servers. Each user's roster and preferences are stored on the server. A user can log on from any client and access user preferences, subscription lists, and messages. The Jabber IM system is very different from existing instant messaging services in several significant ways:

- Jabber is based on the Extensible Markup Language (XML), the universal format for structured documents and data on the Web. This provides an amazing amount of flexibility and scalable.
- Jabber uses a worldwide distributed network, utilizing many interconnected servers. This almost ensures uptime in the network and gives users many options for connecting to the network. For example, if one server is down for any reason, it is a simple matter to move to a new one.
- The Jabber code is open source.
- Jabber has a modular, extensible architecture, allowing the easy creation and integration of new features, especially those that address specific needs. Each new feature is added as a module instead of being added to the core server, so the server maintains its stability while still allowing for new features to be added. (Changes to the server itself are rarely ever needed, but they have been known to happen: One server that we built for a customer was almost unrecognizable as a Jabber server at the time we built it, but some of its features are now quite common.)[7]

2.2 Jabber’s Architecture and Main Concepts

Jabber technologies use a client-server architecture, not a direct peer-to-peer architecture as some other messaging systems do. It is almost identical to that of email. Each user has a local server which receives information for them. The various local servers transfer the messages among themselves for delivery to users. We could see in figure 5, the architecture is similar to the e-mail system where each client is connected to a local server and every communication from or to this client passes this server. User accounts and user data, such as contact lists and preferences, are stored on the local server. User identities are unique for each server by a unique address, like Alice@Jabber.org. Alice can not only contacts Bob who shares the same Jabber server, but also exchanges information with other servers user Cindy or friend David different IM systems over the Internet through a component called Etherx, which takes care of translation between different servers or messaging system. A transport is a special server with the sole purpose of bridging from Jabber to other services (, ICQ, AIM, MSN, etc.). When a user logs onto Jabber, a thread is created in the
transport to handle all communications to and from that user. In addition, a separate thread is created in the transport for each service that the user is subscribed to. [8]

![Jabber Architecture Diagram]

**Figure 5  Jabber Architecture**

**Jabber IDs**

Every entity, for instance, a server, a component, a user connected with a client, is identifiable by a *Jabber ID* (or JID). Each *Jabber ID* contains a set of ordered elements as domain, node, and resource in the following format: user@host/resource or [node@]domain[/resource]. Domain name is the primary identifier. It represents the Jabber server to which the entity connects. The node is the secondary identifier. It represents the “user”. Resource is an optional third identifier. Used to identify specific objects belong to a user. Resources enable a single user to maintain several simultaneous connections.

**Jabber Server**

A Jabber server plays three primary roles:

- Handling client connections and communicating directly with Jabber clients
- Communicating with other Jabber servers
- Coordinating the various server components associated with the server

Jabber servers are designed to be modular, with specific internal code packages that handle functionality such as registration, authentication, presence, contact lists, offline message storage, and the like. In addition, Jabber servers can be extended with external components, which enable server administrators to supplement the core server with additional services such as gateways to other messaging systems.

**Jabber Client**

All clients will be able to:

- Communicate to the server through TCP sockets
- Parse well-formed XML
- Understand the Message data type
- Express presence (online/offline/unavailable) information to the server and understand incoming presence data
2.3 Jabber Protocol
The main protocol of Jabber is the IETF XMMP (EXtensible Messaging and Presence Protocol) Draft. It allows client-to-server and server-to-server communications. The different entities of the Jabber architecture pass data to each other using XML streams, which is essentially the exchange of data in the form of XML fragments in “streaming” mode over a network connection. Jabber manages to cover all the extensive needs in an IM system with these three basic elements, through clever use of XML namespace:

- `<message>` (client to client conversation): protocol for message
- `<iq>` (query message): protocol for presence feature
- `<presence>` (update a client’s availability): protocol for session establishment and roster management

And there are two special namespaces:

- `<error>` (standard error-handling specification)
- `<x>` (allows developer to extend the base protocol)

2.4 Client/Server Interactions
The following segments show a detailed description of communication between clients and servers in Jabber.

**Authentication / Logging In**
If a registered user wants to contact with other users, it must be authenticated at a server. When a request arrives, after checking the session for this user exists or not, the authentication or registration procedure starts. Jabber uses a one-way secure hash function (or, SHA1) to authenticate the user. This function creates a value by hashing the session ID and the user password. The server verifies that the correct hash value was returned by the client. User information with the correct authentication can be stored in a flat file or using a Lightweight Directory Access Protocol (LDAP) service.

**Establishing session**
After some pre-conditions are met, such as Stream Authentication and Resource Binding, the client tries to
contact with the server. Once contacted, the client sends a Session Request message—an XML stream—to the server. If no error occurs, the server grants a session over an open socket, and informs the client that the session has been created. This XML stream is kept open during the lifetime of the session, and all communications between the client and server are sent and received over this socket.

**Sending and Receiving Messages**

After a client has established a connection, it is able to send and receive messages to and from other clients. The source client produces a XML-standard and addresses it to the destination client. The source client’s server is in charge of delivering this message to the destination client (in case it is connected to the same domain server) or to forward this message to the destination’s server. The main fields that a message includes are: *Intended Recipient* (obligatory), *Message Type*, *Message Body*, *Message Subject* and *Conversation Thread*.

**Rosters**

In Jabber, one's list of contacts is called a roster. A roster is stored by the host so that a user may access roster information from any connected resource. It contains subscription information for the client's account, including the user's nickname and contact list. It is shared between the client and server. Roster management includes the following 3 basic aspects:

- Receiving One's Roster on Login — upon connecting to the host, a node should request the roster (however, because receiving the roster may not be desirable for all resources, e.g., a connection with limited bandwidth, the node's request for the roster is optional.
- Adding a Roster Item — at any time, a node may add an item to its roster. The host is responsible for updating the roster information in persistent storage, and also for pushing that change out to all connected resources for the node. This enables all connected resources to remain in sync with the host-based roster information.
- Deleting a Roster Item — at any time, a node may delete an item from its roster. Note: as with adding a roster item, when deleting a roster item the host is responsible for updating the roster information elsewhere.

**Subscriptions**

A subscription is a request to send and/or receive presence information from contacts each time you log onto Jabber. Presence subscriptions are managed by the server and are stored in the roster. When a user logs onto the Jabber server, it sends an update on that user's presence to all of the people on the user's subscription list. There are 5 basic Subscription States:

- None
- To — the User is subscribed to the Contact's presence (only)
- From -- the User is an “observer” in the Contact's roster (only)
- Both -- Both the Jabber User and the Contact are subscribed to each other's presence
- Initial —neither has knowledge of the other's existence (or state in the other party’s roster)

**Presence**

- *Presence update* — Informs people of your current presence state. The *presence agent* updates the presence status on behalf a user by using the presence type: Available. In addition, the *status* attribute is used to describe the specific location in which the user is located.
Presence subscription management – Allows people to subscribe to another user's presence and control who has access to their own presence. A client sends presence information to the server, which then sends it to all of the people on the roster who are subscribed to the client's presence. A number of attributes are optional:

- **Status**: free-form text describing a user's presence (i.e., gone to lunch). Basic presence statuses:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>The normal situation where the user is available. If the show(online) tag is not present, this is used as default value.</td>
</tr>
<tr>
<td>Chat</td>
<td>Similar to normal, but in addition indicates that the user is open to the conversation.</td>
</tr>
<tr>
<td>DND</td>
<td>Stands for “do not disturb” means that the user is available but does not want to be disturbed. However, messages are immediately sent to the user.</td>
</tr>
<tr>
<td>Away</td>
<td>The user is temporarily away from the client.</td>
</tr>
<tr>
<td>Xa</td>
<td>Stands for “extended away” and indicates that the user will not return to the client for some time.</td>
</tr>
<tr>
<td>Offline</td>
<td>The user disconnected from the server. The communication is not available to this user.</td>
</tr>
</tbody>
</table>

Table 1 Possible <show/> Values

3 Comparison of three popular IM Software

**ICQ**

**Description**: pronounced “I seek you”—was one of the first IM products, perhaps most well-known on the market, and is still very popular today. ICQ has good functionality when it comes to presence. Users are shown as logged on or logged off, and it is possible for each user to set his own presence status, for example “out for lunch”, “busy with a meeting”, etc. From being a way of exchanging short text messages between computers, the service has been extended to include functionality such as file transfer, SMS messaging, tele-conferencing and multi-user gaming. ICQ was bought by America Online for $400 million in 1998.[11]

**Features:**
- Organization of contacts in different groups
- Conferencing with multiple contacts at the same time
- Sending of any size files directly between users
- Sending of longer messages by SMS or email
- Central stored contact list
- Voice message, ICQ phone-talk for free

**MSN Messenger**

**Description**: MSN Messenger is Microsoft’s instant messaging client, and is the most popular instant messaging service. With Microsoft’s domination on the personal computer market, the popularity of MSN Messenger should come as no surprise. Further on, the tight integration between Hotmail, .NET Passport and MSN Messenger accounts, serve to consolidate MSN Messengers position in the IM market. MSN Messenger also serves as a distribution channel for content. [12]

**Features:**
· Make worldwide phone calls
· Conferencing with multiple contacts at the same time
· Sending of files directly between users
· Talk instead of type, using a microphone connected to the PC
· Messages by SMS
· Synchronous work on documents (using NetMeeting component)
· Videoconferencing via NetMeeting

Yahoo! Messenger

**Description:** Yahoo! Messenger was released June 1999 and is similar to MSN Messenger in that it, in addition to instant messaging, serves as a distribution channel for content: domestic news, sport results and the weather forecast for the user’s country is available. Further on, airplane tickets can be booked and the Yahoo! Search engine can be accessed. Yahoo! Messenger has a Personal Information Management (PIM) system as well. Data is stored serve-side, and can be accessed from a web-browser or the Yahoo! Messenger client.

**Features:**
· Many entrainment service: LAUNCHcast Radio, games, IMViroments
· Advanced personalization: Audibles, abundant emotions, Yahoo!Avatars
· connect to My Yahoo! more information like stock, weather, news
· Web cam, Conference

**Comparison**

<table>
<thead>
<tr>
<th>System</th>
<th>Advantage</th>
<th>Disadvantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jabber</td>
<td>open-source XML-based decentralized server architecture</td>
<td>More or less a concept, not complete and so popular</td>
</tr>
<tr>
<td>ICQ</td>
<td>send offline message provides great functionalities centralized server architecture</td>
<td>not an open-source protocol security aspect synchronous communication leads to slower many operations on client-side</td>
</tr>
<tr>
<td>MSN Messenger</td>
<td>asynchronous protocol(SIMPLE) many functionalities centralized server architecture</td>
<td>not an open-source protocol Integrates heavily with Windows target for viruses/hackers</td>
</tr>
<tr>
<td>Yahoo! Messenger</td>
<td>centralized server architecture excellent integration with Web powerful chat room</td>
<td>not an open-source protocol many distractions for user: games, advertising, radio weak security features</td>
</tr>
</tbody>
</table>

**Table 2 Advantages and Disadvantages of 4 IM systems [13]**

**4 Conclusions**

Combining two key technologies - instant message delivery and presence information – is the attractive...
features of IM. So IM systems have become very popular over the past years and the number of people who
use ICQ, AIM or the Microsoft Messenger has risen enormously. A presence and instant messaging system
can be used as an integrated communication platform. The presence service can help a user to decide which
kind of communication service is the best to reach another user. An intelligent instant messaging server can
also route messages according to the user’s preferences.

The most attractive part of Jabber is its open source platform. The Jabber protocols are free, open, public,
and easily understandable. It offers support for interoperability Security and Scalability. Meanwhile the
client implementation is Simple. But the disadvantage is its heavy XML coding and not so widespread and
popular. It is more or less a concept but without powerful occupation of the market. Compare to Jabber, ICQ,
MSN and Yahoo, each of them has their own widespread protocol, special customer-oriented services and
richer set of functionalities. More and more personality express make them millions of young people and
lots of company choose them as the first choice.

Now, instant messaging services are going to the mobile domain.[14] The basic benefit of mobile instant
messaging is still the “see before you connect” aspect, but mobility adds a number of features to make it
even more attractive to the user. Along with the progress of wireless transmission and the volatile increase of
wireless network users, the demands of using handset and combining the presence status of friends online to
delivery message on the mobile network as well as using handset and IP network to execute over-platform
integrated information transfer are all emerged gradually. We could see a much broader foreground of IM
system in the coming future.

5 References

0764549537
[3] Rochus Schreiber and Ragnar Eggen, Why wireless instant messaging may be the key to migrating
mobile customers to 3G service. http://www.accenture.com
[4] Ramiro Liscano Professor, Presence and Awareness Services, SITE U. Of Ottawa
University of Applied Science Stuttgart – Hochschule der Medien, June 4, 2004
[10] RFC-3920 XMPP Core