Presence and Mobile Instant Messaging

Ajith Kannath
Computer Science Department
University of California, Irvine
akannath@uci.edu

Sandeep Katumalla
Computer Science Department
University of California, Irvine
skatumal@uci.edu
1. Introduction

The rapid development of many modern communication technologies have given users, many more options for communicating in their workplace and personal life. Of these new communication technologies, instant messaging is one of the most proliferating communication technologies adopted. The rise in popularity of instant messaging in recent years have led to its acceptance as a valuable method of communication in academic and professional sectors. Instant messaging protocols have enabled users to interact in a faster way as compared to protocols like e-mail while avoiding any potential costs as opposed to voice telephony. Instant messaging is looked upon as the major killing application in the field of IP-telephony.

In telecommunication networks, presence information is defined as a status indicator showing the willingness and ability of a potential communicator to communicate. The client application on the user's side contacts a presence service and provides presence state via network. At the server end it is stored in his personal availability record and is distributed to other users to convey his availability for communication. Modern instant messaging clients use various user availability states such as "free for chat", "busy", "away", "do not disturb" etc. to portray the users current mood, location or free text status. Presence shows the availability state of a user (and maybe much more information about his current state, location etc) and any other user can make use of this data to decide whether to initiate a conversation with that user.

Instant messaging (IM) is a form of real-time text based communication between two or more people with personal computers or any other devices using shared clients. Instant messaging goes hand-in-hand with presence in the sense, a user needs to know the presence information of another to initiate a messaging session with him. IM is considered as one among the various 'online chat' protocols and the specific thing about IM is that it facilitates connection between a specified list of known users called 'buddy list', 'friend list' or 'contact list'. It differs from e-mails in the sense that IM is real-time communication. It allows the user, advantages of both email systems, where communication is free and telephonic systems where communication is real-time.

Modern companies try to combine different solutions with instant messaging and produce combined and highly potential applications. We will be discussing many of the current instant messaging architectures in the next section. It is also possible to save instant messages for a later reference and many current instant messaging/chat systems support this feature. Each modern IM generally provide its own client which can be a separately installed software or a browser based client. Also, lately there are many third-party client software applications that will connect with most of major IM services. Modern systems hardly distinguish between IM and MIM (Mobile Instant Messaging) so we will be using both terms interchangeably with the same meaning implied. Also MIM is now available in the form of many applications in many embedded operating systems as Windows Mobile, Blackberry OS, Symbian OS, Android OS etc. Many modern IM systems even allow to incorporate microblogging functionalities within them enabling the users in the 'friends list' to comment on the status message of a particular user. A recent example for such an application is 'Google BUZZ'.

We are planning to design and develop a SIP (Session initiation protocol) based presence and instant messaging protocol. So as a first step we have done a wide survey of all the existing distributed or centralized protocols which deal with presence and messaging. We have analyzed many IM systems which are currently in market, and the protocols which they follow. In the next section we have tried to comprehensively summarize the architectures of all current IM systems. In section 3, we will discuss the design and architecture of IM and presence system that we are planning to develop. In the section 4, we will discuss some future directions for our application and give some concluding remarks.
2. Existing Presence and Messaging protocols

**Extensible Messaging and Presence Protocol (XMPP):** XMPP is an open standard communications protocol for message oriented middleware based on XML. The XMPP network which is a free and open source software uses a client–server architecture that is decentralized and hence there is no central authoritative server. It was originally developed for near real time extensible instant messaging, presence information and contacts maintenance. Owing to its extensibility, it led to its widespread use in VoIP (Voice over Internet Protocol) and file transfer signaling. The unique feature of XMPP is that it uses an open systems approach of development and application, which allows anyone to implement an XMPP service that can interoperate with other organizations' implementations. The main strengths of XMPP lie in the areas of decentralization, maintaining open standards, security features and its flexibility. A key security feature of XMPP is the ability to isolate XMPP servers from a public XMPP network. Also, in order to encourage channel encryption, the XMPP standards foundation also acts as an intermediate certification authority which offers free digital certificates to XMPP server administrators. Moreover, custom functionality can be built on top of XMPP services to extend interoperability. Another unique feature of this protocol is that anyone may run their own XMPP server on their own domain with the standard TCP port of XMPP which is 5222. Every user on the network has a unique ID. To avoid requiring a central server to maintain a list of IDs, the ID is structured similar to an e-mail address with a username and a domain name (or IP address) for the server where that user resides, separated by an at sign (@). This allows users wanting to log-in from multiple locations, by specifying a resource that identifies a particular client as belonging to the user. This may be included in the ID by appending a slash followed by the name of the resource.

**Real-time Transport Protocol (RTP):** This is Another important protocol that is relevant to instant messaging which standardizes video and audio packets for delivery over IP networks. It is mainly used in conjunction with RTCP or Real-time control protocol, which monitors transmission statistics and QoS, along with the synchronization of multiple streams. Designed for end to end real time transfer of stream data, it also provides for jitter compensation and detection of out of sequence packets. It is widely considered to be the primary standard for multimedia transmission over IP networks. The Transmission Control Protocol (TCP) is not normally used in RTP application because TCP favors reliability over real time transfer. As a consequence of this, a majority of the RTP implementations are built on the User Datagram Protocol (UDP). Other transport protocols specifically designed for multimedia sessions are SCTP and DCCP, are not in widespread use. It works by establishing an RTP session for each multimedia stream. A session is made up of an IP address with respective ports for RTP and RTCP. There are separate RTP sessions maintained for audio and video streams, which give the receiver the choice of deselecting one of the streams. It is based on the principle of application level farming which mandates that the information required by a specific application's needs are not present in the generic RTP header. Moreover, RTP defines separate profiles and payload formats for each class of application such as audio or video. Profiles define the individual codecs used to encode the payload data, which is in turn accompanied by several specifications for payload formats. The aim is to make a network based system which will include other protocols along with RTP such as SIP (Session initiation protocol).

**The Microsoft Notification Protocol (MSNP):** Developed by Microsoft, MSNP was primarily designed for the .NET messenger service and clients attached to it, and was also made flexible enough to be adopted by third party clients such as Pidgin and Trillian. MSNP consists of a series of commands sent between the client and the server which establish a connection on request from a user and passes the data between two endpoints in a conversation. Subsequent versions of MSNP have incorporated features such as VoIP (Voice over Internet Protocol) and video streaming. The MSNP has a breaking point version of the protocol that supports high performance simulations allowing a 10GB blade to generate over 2.4 million concurrent TCP flows or in other words, about
5 million people to communicate through the messenger server at any given time. It works with conditional requests and uses SSL for accurate authentication transactions. At the advent of MSNP8, the service introduced a different authentication method, which sent authorization requests to Microsoft Passport's secure servers which in turn returned a challenge string. Windows Messenger uses MSNP8 as its standard version, including 4.7 upto the latest version 5.1. This protocol also supports webcam and voice capabilities. The MSNP9 version provided support for the D type messages which were data messages used for transferring picture message, files, emotions as well as frame by frame web cam video as opposed to the standard practice of streaming video, as well as better network address translation(NAT) traversal for file transfers. The latest version of the protocol which is the MSNP15 was introduced along with Windows Live Messenger 8.1. It was based on a different authentication mechanism called RPS(Relying Party Service). In addition to a new authentication mechanism, Microsoft also made more of the properties of the user roaming. Which meant that the user's display picture, and in the future personal status messages, will be the same wherever the user signs in. Furthermore, support for user locations has been added to the Personal Status Message, although this feature was later removed from the Windows Live Messenger 8.1 client.

.NET Messenger: One of the first instant messaging and presence systems was the .NET Messenger Service or the MSN Messenger Service developed by Microsoft, which used the Microsoft Notification Protocol. Also called the Windows Live Messenger, it is currently designed to work with the latest versions of Windows, Blackberry OS, iOS, Java ME,S60 and the Zune HD. It offered many first time features such as the Album viewer, selective offline status, integration with social networks as well as off line messaging. The latest version is Windows Live Messenger 2009. It was proposed to have features related to multi-person audio/video chat, and they are also attempting to create interoperability with AIM/XMPP/ICQ. A new "Groups" feature was also introduces that allows users to create a continuous group conversation between select contacts, newly redesigned status icons, a new "Favorites" category in which once can place his/her favorite contacts for easy access to them, photo sharing utilities that allow contacts to quickly and easily browse photos together, and a "What's New" section at the bottom of the contact list to outline recent contact updates. Another important feature was that it is the first version of Windows Live Messenger to use the standard window frame on Windows Vista in accordance with the user experience guidelines.

Google Wave: Perhaps the latest in instant messaging protocols is the Google Wave Federation Protocol is an extension of XMPP that is used in Google Wave. It is designed for near real-time communication between the computer supported cooperative work wave servers. It has incorporated some inherited features of Extensible Messaging and Presence Protocol (XMPP) such as the discovery of IP addresses and port numbers, using Domain Name System (DNS) SRV records, and TLS authentication and encryption of connections. The XMPP transport encrypts operations at a transport level and hence only provides cryptographic security between servers connected directly to each other. An additional layer of cryptography is used to provide end-to-end authentication between wave providers using cryptographic signatures and certificates, allowing all wavelet providers to verify the properties of the operation. Therefore, a downstream wave provider can verify the authenticity of wavelet providers. This addresses the situation where two users from different, trustworthy wave providers are participants of a wavelet that is hosted on a malicious provider. The protocol requires each participant to sign its user's operations with its own certificate which will be later evaluated by the participants. This is to stop malicious hosts from altering or spoofing the contents of the messages from the user of other services. All the signatures and verifications are done by the wave providers and not the client software of the end users. All waves and wavelets (child waves) have a globally unique wave id consisting of a domain name and an id string. The domain name is used to identify the wave provider who originated the wave. Waves and wavelets are hosted by the wave provider of the creator. Wavelets in the same wave can be hosted by different wave providers but the user data is not federated; i.e., not shared with other wave providers. It also provides for the private reply of wavelets, of which other participants have no knowledge or access. If a private wavelet is sent between users on the same wave provider, it's not
federated regardless of where the parent wave is hosted.

**Skype Protocol:** One of the most recent names to emerge in the domain of instant messaging and VoIP is Skype. Skype uses a proprietary Internet telephony (VoIP) network based on peer-to-peer architecture. The protocol has not been made publicly available by Skype and official applications using the protocol are closed-source. The Skype network's unique feature is that it is not interoperable with most other VoIP networks without the required licensing from Skype. Skype was the first peer-to-peer VoIP network, which meant that it requires minimal centralized infrastructure. Its user directory is decentralized and distributed among the clients, or nodes, in the network. The network contains three types of entities namely supernodes, ordinary nodes, and the login server. Each client maintains a cache of IP address and port numbers of reachable supernodes. Any unprotected client with good bandwidth and adequate processing power can become a supernode, in effect opening up the possibility of it being used by Skype to convey network traffic. Supernodes relay communications for clients behind firewalls and Symmetric NAT. The disadvantage of this is that it puts an extra burden on those who connect to the Internet without NAT, as Skype may use their computers and Internet connections to relay other users' calls. After a Skype client is connected, the next step is to authenticate its username and password with the Skype login server. There are many different Skype login servers using different ports a list of which is hardcoded in the Skype executable. On each login session, Skype generates a session key from 192 random bits. The session key is encrypted with the hard-coded login server's 1536-bit RSA key to form an encrypted session key. Skype also generates a 1024-bit private/public RSA key pair. An MD5 hash of a concatenation of the user name, constant string and password is used as a shared secret with the login server. On the login server side, the plain session key is obtained by decrypting the encrypted session key using the login server's private RSA key.

**Yahoo! Messenger:** Another popular instant messaging protocol is the Yahoo messenger protocol. Like most other cutting edge instant messaging technologies, this protocol too supports many features beyond just messaging, such as off-line messaging, file transfer, chat, conferencing, voice chat, webcams and avatars. The purpose of the YMSG protocol is to provide a language and series of conventions for software communicating with Yahoo!'s Instant Messaging service. In essence YMSG performs the same role for IM as HTTP does for the World Wide Web. Unlike HTTP, however, YMSG is a proprietary standard, aligned only with a single messaging service provider. Rival messaging services have their own protocols, some based on open standards, others proprietary, each effectively fulfilling the same role with different mechanics. One of the fundamental tenets of instant messaging is the notion that users can see when someone is connected to the network — known as 'presence'. The most important feature of Yahoo's protocol is that it uses the mechanics of a standard internet connection to achieve presence which is the same connection it uses to send and receive data. In order for each user to remain 'visible' to other users on the service, signaling their availability, their Yahoo IM client software must maintain a functional, open, network connection linking the client to Yahoo!'s IM servers. As some organizations block communication on the port used by Yahoo! IM, either because they choose to whitelist certain types of internet usage (only web surfing and email, for example) or because they seek to blacklist instant messaging services, Yahoo provides an alternative route for connecting to their service which mimics the HTTP protocol used by the World Wide Web. However, since HTTP has no inherent sense of a persistent connection, this protocol instead relies on the client frequently contacting the server in order to approximate the sense of a connection required to give each user presence on the IM network.

**Jingle:** Another peer to peer session control protocol is Jingle which is again an extension of XMPP. Here, for the first time multimedia streams are delivered using the Real-time Transport Protocol (RTP), for services such as VoIP and video conferencing. The Zephyr protocol is another Unix based application suite which is a conglomeration of multiple separate programs working together towards a common cause. Along with the IRC, Zephyr was the first widely used IP based instant messaging system. Today, Zephyr is rarely used, and the few places it is still used is in
university environments, but has largely been replaced by the latest applications such as XMPP. From the current project's point of view, which is to develop a presence and instant messaging application on mobile devices, a set of specifications called Wireless Village is relevant. According to Wireless village specifications, the protocol must make use of existing internet and web technologies. Also, the protocol uses XML to represent protocol data that is being transmitted during an instant messaging session. Other useful standards in this space include the Multipurpose Internet Mail Extensions (MIME) for registering the format of these protocol messages.

Lastly, we conclude with TOC or Talk to OSCAR protocol which was used mainly by third party AOL instant messenger clients. TOC works as a wrapper for the OSCAR protocol. From a high level view, the TOC server was similar to an OSCAR client listening to a socket and translating messages between two dissimilar protocols.

3. Implementation of IM and Presence using SIMPLE

We have seen many end to end instant messaging sessions in the last section and have gone through many of their properties. Many of the common users use one or the other of these IM systems, but many of them have insufficient support for multimedia and no open standard for intercommunication between them. So it makes sense to unify several major communication services and the best protocol of choice would be Session Initiation Protocol (SIP). IETF has established SIP Instant Messaging and Presence Leverage Extensions (SIMPLE) workgroup to develop a robust, flexible and secure IM & P protocol based on SIP.

3.1 Basics of SIP, IM and Presence

SIP is the standard protocol for initiating, modifying or terminating the communication sessions between users. In SIP based networks, the entities are: 1) User agents – the end-users of the application and 2) Network servers – proxy server, redirect server, location server etc. A user agent initiates a session by sending a SIP INVITE to SIP proxy. It also sends a SIP REGISTER message to the location server with its address in the contact header. Apart from these messages, there are SIP SUBSCRIBE and SIP NOTIFY for extending SIP event notification function. Please refer the following figure for a general architecture of IM & P.

![Figure 1](image-url)
The general interaction between different modules can be generally classified into four:

1) **Subscription**: An end-user sends SUBSCRIBE message to the Presence Server and the server authenticates the subscriber based on the format of his message.
2) **Publication**: A client sends presence information to server for distribution.
3) **Notification**: The Presence Server delivers the published data to other subscribers in the network.
4) **Message Sending**: This module establish session between subscriber and client for sending messages.

### 3.2 Architecture of IM & Presence based on SIMPLE

The SIP location servers already have presence information in the form of registrations. The SIP servers are capable of routing requests from any user in network to a server that holds registration state for any user. The basic components of a SIMPLE architecture are:

1. **Presence user agent (PUA)**: The entity that manages presence information of a presentation.
2. **Presence Server (PS)**: A physical entity acting as presence agent or proxy for SUBSCRIBE requests.
3. **Watcher**: A logical entity requesting for presentation.

The working of SIMPLE protocol is as follows. Any user agent (PUA) sends out a REGISTER request to publish its presentation. The request reaches the registrar through a proxy and gets registered in the server. Now consider the case of a watcher who had not subscribed for presentation information of PUA, sends a SUBSCRIBE for PUA to server. The presence server forwards this request to PUA and PUA replies with its willingness to share the information and subsequently, the PS will send NOTIFY information through proxy to the watcher.

### 3.3 Implementation

To implement the SIMPLE architecture, the JAIN-SIMPLE design model is used as shown in the figure below:

![SIMPLE Architecture Diagram](image)
The components shown in the graph, (Client Application, SIMPLE components, SIMPLE provider and SIP stack) operate as explained in section 4.2. Each SIMPLE components register with SIMPLE provider to communicate with underlying SIP stack. The SIP server that we are planning to use is the 'Ondo SIP server' which is there on all operating systems including Windows and Linux. The client application is developed in Java on Android platform.

3.4 Extensions and future plans

Most of the modern presence and message systems support additional multimedia functionalities apart from plain textual chat. These additional functionalities include video and voice chat, file (image, audio, video or document) transfer, sending the off-line text message as an email etc. After the basic functionalities are up and working, our next concentration will be to add one of these additional features and make the application more and more usable in day-to-day life.
4. References


[2] SIP for Presence and Instant Messaging, Helsinki University of Technology, Networking Laboratory


[6] Instant Messaging and Presence Technologies for College Campuses, Samir Chatterjee, Tarun Abhichandani, Haiqing Li, Jongbok Byun, and Bengisu Tulu

[7] Comparison of SIP and H.323 Protocols, Ilija Basicevic, Miroslav Popovic, Dragan Kukolj

[8] Instant Messaging and Presence on the Internet, Harald Alvestrand


