RESEARCH ARTICLE

A Novel Design of Instant Messaging Service Extended from Short Message Service with XMPP

Heng-Te Chu
Dept. of Information Networking Technology, Hsiuping University of Science and Technology, Taichung, Taiwan
henrychu@hust.edu.tw

Lili Hsieh
Dept. of Information Management, Hsiuping University of Science and Technology, Taichung, Taiwan
lily@hust.edu.tw

Wen-Shiung Chen
Dept. of Electrical Engineering, National Chi Nan University, Pui, Nantou, Taiwan
wschen@ncnu.edu.tw

Abstract – Instant messaging (IM) services have grown dramatically in recent years. In telephone networks, short message service keeps dominating mobile data services. As the rise of nomadic need, to bridge both messaging services is becoming a promising niche. In this paper we propose an infrastructure, based on the XML-based protocol, XMPP, to simplify the interconnections and enable the legacy handsets for instant message service. Although there is another competitive approach, SIMPLE, based on SIP, we will analyze how the XMPP is to outdo SIMPLE.

Index Terms – Instant Messaging (IM), Short Message Service (SMS), XML, XMPP, SIP, SIMPLE.

1. INTRODUCTION

Instant messaging (IM) service and presence technology [1] enable people to exchange messages more time-effectively than e-mail, and to be able to know the availability of other users. Meanwhile, short message services (SMS) [2, 3] are still the most popular data service for people with mobile handsets. With the always-on nature of handsets, people get used to get messages instantly anywhere at any time. Although, nowadays, IP-based applications can communicate with a short message service center (SMSC) [2, 3] in the short message peer-to-peer protocol (SMPP) [4], SMPP is obscure for the need of fast deployment. Since SMPP is in binary data format, instead of human-readable text format. Most application developers view it as a technical barrier to join.

On the other hand, major Internet instant message service providers (e.g., AOL, MSN, Yahoo, ICQ, and so on) dominate their own communities by using their own proprietary interface to secure their investment. Their proprietary protocols block interoperability and also raise another barrier, too. The Extensible Messaging and Presence Protocol (XMPP) is an open XML protocol for near-real-time messaging, presence, and request-response services [5-8]. Since XML [9] is in human-readable text format, it is open, flexible, portable, and simple to create and read. Most modern programming languages do support XML. Through XML, it is easy to build a gateway to interoperate with non-XMPP messaging system, like SMSC, MSN, AOL, Yahoo, ICQ, and so on. An XMPP gateway is a special-purpose server-side service whose primary function is to translate XMPP into the protocol used by a foreign (non-XMPP) messaging system, as well as to translate the return data back into XMPP. Therefore, the gateway can be used to cover up the complexity of SMPP and other proprietary protocols, and keep the same XML-based interface to outside world for simplicity.

In this paper, we are going to first brief the technical backgrounds about XMPP and SMPP respectively in section 2 and 3. Then, we will show some common practical but proprietary integration approaches used by many independent service providers. The rest of this paper will demonstrate how we integrate both IM and SMS in XMPP manner, analyze why we choose XMPP, rather than SIP, and conclude this paper.

2. XMPP ARCHITECTURE

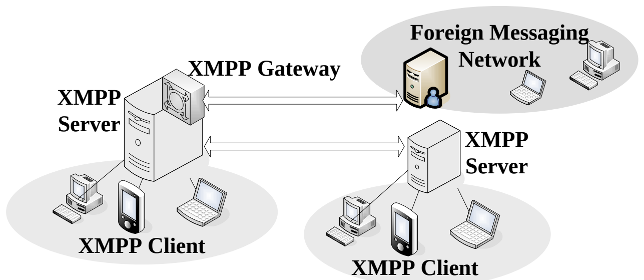

Figure 1 Overview of the XMPP architecture.

The high level overview of XMPP architecture is shown in Figure 1. The XMPP server manages connections or sessions for other entities, in the form of XML streams to and from authorized clients, servers, and other entities.

The XMPP server is also responsible for storing client data and routing messages among entities. The XMPP addressing format is like, user@domain/resource, which is quite similar to e-mail address, except the resource part. The domain part guides how to relay messages from one domain to another. The resource part indicates a particular message delivery endpoint for a user, e.g., Joe@A.com/home or Joe@A.com/mobile, as shown in Figure 2. All XMPP data are delivered to resources. As shown in Figure 3, an XMPP server takes charge of parsing incoming XML streams and routing outgoing XML packets to the best and/or preferred client's resource available for a user.

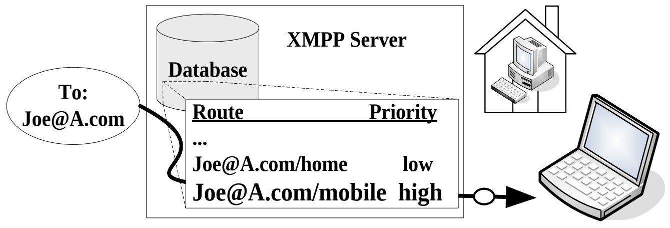

Figure 2 Routing to the best and available client

Instant message delivery needs to determine where to deliver across network, and when to reach as soon as recipients becoming available. Figure 3 shows that the XMPP server designs a packet queue to store and forward the packets from the XML parser. Without scratching its own queuing mechanism, we may make use of existing message queue technologies or products.

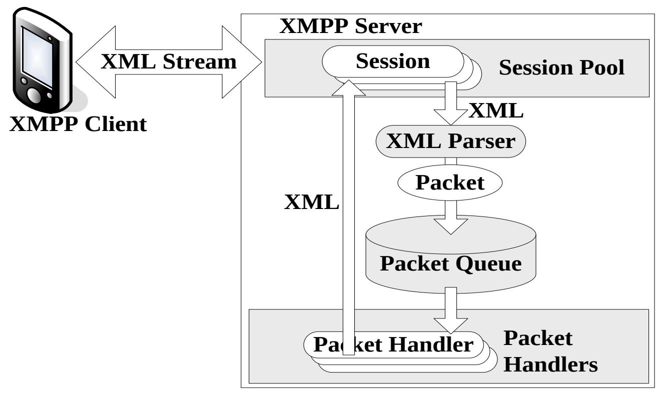

Figure 3 Basic functional modules of the XMPP server

IM services rely on user presence information to determine best target for a user. Instead of having every user sending their presence to other users, it can go by way of presence subscription, as shown in Figure 4. A client may send a request to the server in order to subscribe presence updates of target clients. The target clients may accept or refuse to reveal its presence update for privacy concern. The XMPP server hosts, organizes and maintains user subscriptions, arbitrates and broadcasts all presence updates.

A user may have many sessions, each with its own presence status (e.g., your PC is off, but your mobile handset is at hand). However, people just care if they can send messages to another person, regardless of what client or device to be. The XMPP will rely on user presence for message routing and store-and-forward delivery, to reach the best available client.

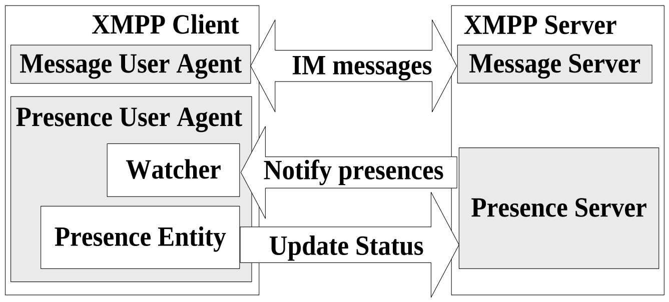

Figure 4 The XMPP controls presence exchanges

Although XML is in human-readable text format, the content itself may be secured independently. The XMPP includes a method for securing the stream from tampering and eavesdropping. This channel encryption method makes use of the Transport Layer Security (TLS) protocol [10] for client-to-server communications, server-to-server communications, or both. Furthermore, XMPP includes a method for authenticating a stream by means of an XMPP-specific profile of the Simple Authentication and Security Layer (SASL) protocol [11]. SASL provides a generalized method for adding authentication support to connection-based protocols, and XMPP uses a generic XML namespace profile for SASL that conforms to the profiling requirements of SASL.

3. SHORT MESSAGE PEER-TO-PEER PROTOCOL

For brevity, we focus only on its support with GSM technology and TCP/IP network, as shown in Figure 5.

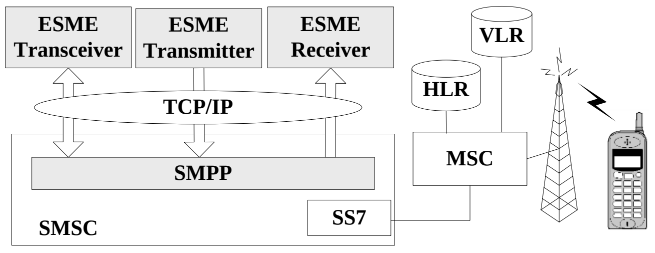

Figure 5 The SMPP interface

The SMPP is based on the exchange of request and response protocol data units (PDUs) between the external short message entity (ESME) [4] and the SMSC over an underlying TCP/IP network connection. Here, the ESME may be a ticket system, an application for headline news, or an advertisement broadcast service.

As shown in Table 1, command length, ID, status, and sequence number are 4-octet unsigned (big-endian) integers. The protocol data unit (PDU) body may contain mandatory and/or optional parameters corresponding to command ID field, defined by the SMPP protocol. Moreover, a GSM short message contains up to 160 7-bit characters or 140 8-bit octets [2, 3]. The 8-bit data are in UCS-2 [12], 16-bit encoding. A conversion between UCS-2 and local character set is required for implementation. Obviously, the SMPP PDU is much harder than XML to compose.

SMPP PDU				
PDU Header (Mandatory)				Body (Optional)
Command Length	Command ID	Command Status	Sequence Number	PDU Body
4 Octets	Length = (Command Length value - 4) octets			

Table 1 An overview of the SMPP PDU format

Usually, a carrier operator creates its own proprietary interfaces to ICP for short message value-added services, with the intention for simplicity, as shown in Figure 6. Those proprietary interfaces can be mainly categorized into the following three types:

1. HTTP + Query String, e.g.,
   http://A.com?ID=123&MSG=hello&…

2. HTTP POST method + XML body, e.g.,
```
<?xml version="1.0" encoding="Big5"?>
<sms>
    <ID>123</ID>
    <MSG>Hello</MSG>
    …
</sms>
```

3. TCP socket + Home-made PDU

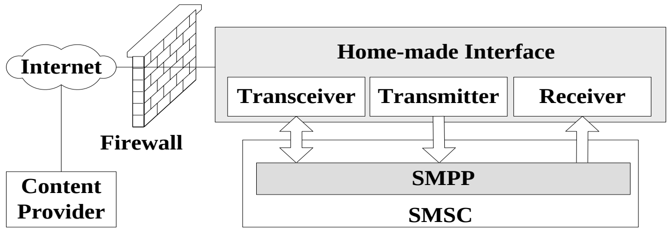

Figure 6 Overview of the operator's proprietary interface

The first type is easy and straightforward at first glance. Obviously, it is vulnerable to attack. Besides, to change the query string formats causes tedious modifications on both operator and ICP sides. Meanwhile, the common delimiter, &, may cause the conflicts while paring and filtering the query string [13]. The second type resolves the disadvantages of the first type. Instead of appending data to URL, it uses the HTTP [14] POST method to hide and convey business complexity. It keeps HTTP URL intact and clear. However, different operators may define different XML tags and formats. It obstructs interoperability. The third type is actually derived from the SMPP directly in certain degree. Inevitably, it comes along with the difficulty as much as the SMPP. Thus, an ICP has to deal with each operator's homemade interface separately. As a result, if there are *m* ICPs and *n* operators, then there will be ($m \times n$) connections among them, as shown in Figure 7.

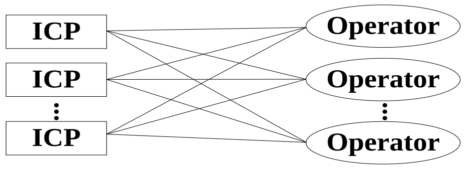

Figure 7 *m* ICPs and *n* operators result in $m \times n$ connections

In addition to the above three types, there is a SIP-based extension for Instant Messaging [15]. However, it simply works like pager messages, i.e., no explicit association between messages. As SIP was originally designed for internet telephone call control protocol between user terminals, the extension discourage the attempt to carry high-volume messages. To put instant messaging and presence service into practice, the SIMPLE (SIP for Instant Messaging and Presence Leveraging Extensions) work group [16] has made many efforts to embed XML-based service messages into a series of proposed standards such as SIP event, presence information data format, privacy and policy, provisioning, federation, optimization, and instant messaging session mode-related protocols.

```
<?xml version="1.0" encoding="UTF-8"?>
  <presence xmlns="urn:ietf:params:xml:ns:pidf"
     xmlns:local="urn:example-com:pidf-status-type"
     entity="pres: john_doe@foo.com">
    <tuple id="sg89ae">
      <status>
        <basic>open</basic>
        <local:location>home</local:location>
      </status>
    </tuple>
  </presence>
```

Figure 8 Sample of presence information

Those protocols embed messages in XML formats. For instance, the presence information [17] is expressed in XML format, as shown in Figure 8. However, to cope with a bunch of standards could encounter significant challenges associated with complexity, scaling, and compliant issues. Thus, our proposed method is based on XMPP, rather than SIMPLE-like solution.

4. THE PROPOSED METHOD

The XMPP gateway plays a key role to realize the function of translation and bridge. The gateway will convert data between the XML formats and the SMPP protocol data units. The embedded ESME will communicate with the SMSC, as shown in Figure 9.

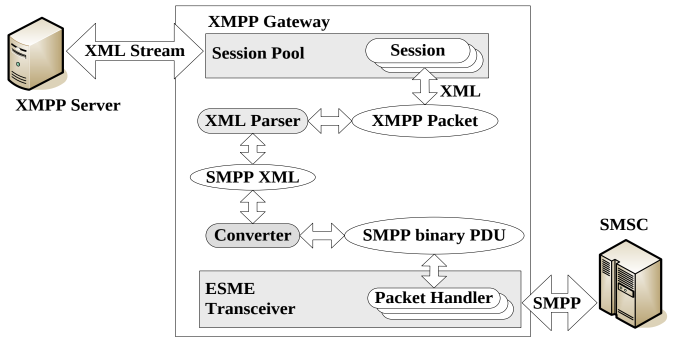

Figure 9 Basic functional modules of the XMPP gateway

The XMPP gateway has only one ESME to communicate with the SMSC. The SMSC is required to create an SMPP account and grant a TCP port, e.g., port number 3456, for the ESME to connect for data transmission, as show in Figure 10. Since mobile handsets are addressed using phone numbers, it is necessary to map between IM address and phone numbers. In addition, the SMPP does not provide session or conversation management. Every short message is a separate event, i.e., a short message does not correlate with any other short messages. To keep track of short messages within an IM session, the session module of the XMPP gateway must create a pseudo XMPP client, as show in Figure 10, in order to act for the corresponding SMS user to login, logout, transmit data, and so on. Besides, while converting from XML packets to SMPP PDUs, the "From address" will be assigned a cyclic serial number and prefixed by an alphabet character, e.g., X000000001 to translate and map from remote IM address to a virtual phone number, for reply path. The SMSC is required to set routing paths for virtual phone numbers as well. When the user replies a short message, its home SMSC will check its routing table for further delivery attempt, as shown in Figure 11.

Similarly, a mobile user, e.g., Joe, can send a short message to a virtual number, e.g., 123, in order to login the XMPP server. As soon as the recipient, e.g., Bob@B.com, is on-line, the XMPP server notifies Joe, as well as shown in Figures 10. Now, Joe may simply reply an SMS to start IM conversation with Bob. In reality, a mobile handset may be unreachable or power off from time to time. Most SMSCs operate in store-and-forward mode for retrial and delivery guarantee. However, short messages may be sent to mobile handsets out of sequence. As a result, it may confuse recipients and collapse sequence-sensitive content delivery.

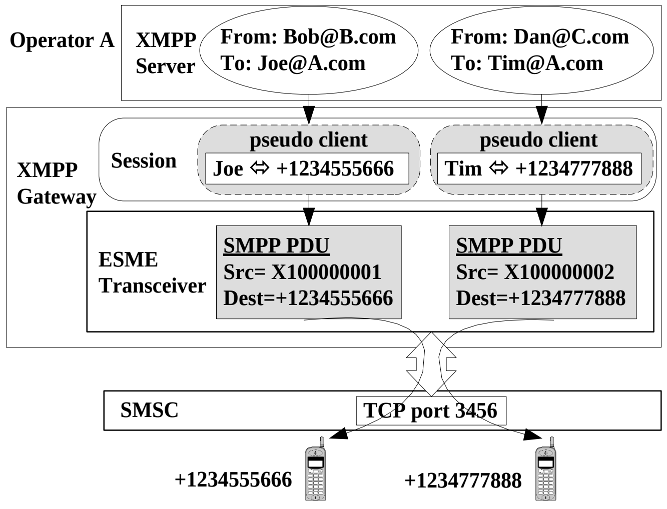

Figure 10 Pseudo XMPP clients for IM session tracking

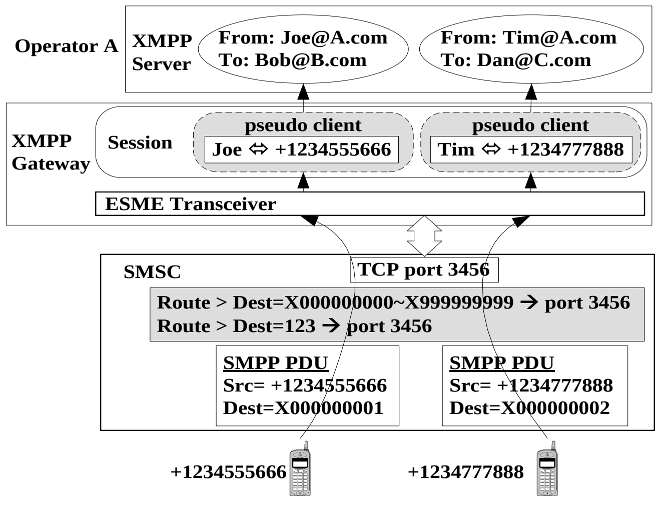

Figure 11 Virtual phone number for reply path

To keep message delivery in order, the ESME of XMPP gateway may utilize the feature of SMPP, delivery receipt, so that the pseudo XMPP client can track message sequences. As shown in Figure 12, the ESME can enable the parameter, registered_delivery, in each submitting SMPP PDUs to request SMSC to report the final delivery status of messages. The pseudo client also needs add an extra tag, <expiration time>, e.g., 60 seconds, to each XMPP XML packets, so that the ESME can ask the SMSC to discard obsolete messages for sake of instant messaging concern.

Therefore, the ESME can feed the final delivery status back to the pseudo XMPP client. The pseudo XMPP client may re-send or control message sending pace to ensure mobile handsets to get meaningful content in right sequence. Therefore, a subscriber of operator A may chat with a subscriber of operator B, as shown in Figure 13, and vice versa. Furthermore, a mobile subscriber may chat with a regular XMPP client as well.

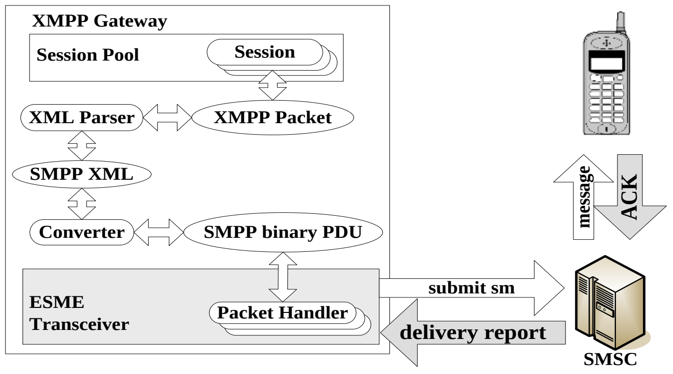

Figure 12 SMPP delivery receipt returns final delivery status

Furthermore, the Internet content provider's application may embed an XMPP client to uniformly communicate with any mobile subscribers as well as any regular XMPP communities to promote services, as shown in Figure 14. As mentioned earlier in Figure 6 and Figure 7, both ICP and operators may suffer from tedious operation and maintenance. By contrast, with the open and unified protocol in the XMPP, as shown in Figure 14, we may relieve both ICP and operators of the burden of interoperability. Like e-mail systems, the XMPP servers may utilize Internet routing mechanism as well as e-mail servers do. Hence there is no need to maintain dedicated or proprietary links among XMPP servers. Virtually, we may consider all XMPP servers as a whole to be an XMPP network, as shown in Figure 14.

Both ICP and operator sides can simply connect to Internet and join the XMPP network, and then, gain the interoperability with the others instantly. Through the IP-based XMPP network, each ICP or operator may just need one IP connection link to build. It reduces the total links significantly from (m×n) down to (m+n).

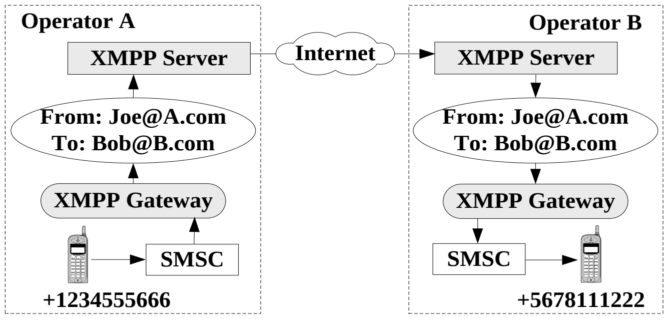

Figure 13 Sending an IM over SMS

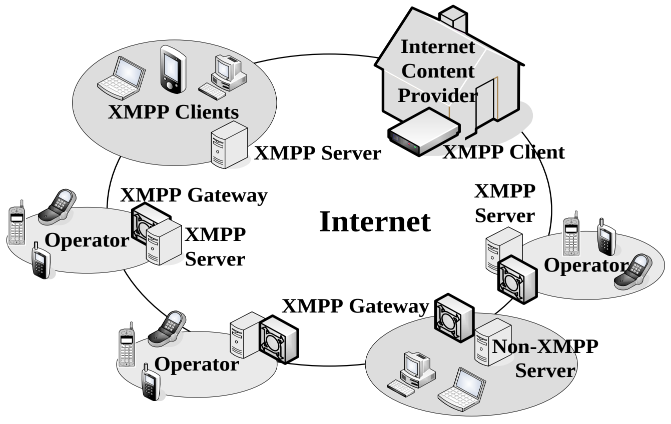

Figure 14 To unify and simplify communication protocols

5. ANALYTICAL COMPARISON

To compare our solution, as shown in Figure 14, with existing proprietary ones, as shown in Figure 6, we list criteria and results in Table 2.

	Proprietary (Figure 6)	Our method (Figure 14)
Interoperability	No	Yes
Portability	Poor	Yes
Extensibility	Limited	Yes
Security	Compromised	Yes
Simplicity	Moderate	Yes

Table 2 Proprietary vs. our method

As mentioned earlier, proprietary interfaces or protocols make interoperability difficult or even impossible. Without a unified protocol, it is hard to port, extend, or upgrade an application from one version to another. Most proprietary protocols are designed to work for closed domains, so that security issues are often compromised. Compared with XML, most legacy protocols, such as the SMPP, are not easy to read and manage. From communication perspective, interoperability among different communities is the most significant and key success that the proposed solution outperforms all existing proprietary interfaces.

Furthermore, the aforementioned SIMPLE is a competitive approach. We would like to compare the differences between SIMPLE and XMPP to show their pros and cons. Firstly, both SIMPLE and XMPP are distributed architectures. SIMPLE, like SIP, works in a peer-to-peer manner and puts intelligence at user terminals. Before transmitting instant messages and updating presence information, the SIP session must be established. During the establishment, there are several SIP request and response messages that take place between user terminals and intermediate SIP servers. Once the SIP session set up, messages are exchanged in-between peers directly. On the other hand, XMPP server handle all connections and all instant messages and presence information, which really appeals to corporate staff for management purpose, e.g., auditing and log. Secondly, both SIMPLE and XMPP use a lot of XML messages, which add significant messaging overhead and consume considerable network bandwidth. Unlike SIMPLE, XMPP is purely an XML-based protocol that enables XMPP to provide more flexibility to business and social applications than SIMPLE. In contrast to XMPP, SIMPLE relies on SIP and other XML-based protocols for much of it functionality, which make it more complex than XMPP. Lastly, Some of SIP-based traffic may utilize UDP, instead of TCP. However, UDP encounters problems and needs extra care during NAT traversal [18]. In contrast, XMPP uses TCP by default. Thus, it can traverse NAT friendly.

	SIMPLE	Our method
Architecture	Distributed	Distributed
Auditing and Log	Partial	Full
Messaging Overhead	Considerable	Considerable
Flexibility	Moderate	High
Complexity	High	Moderate
NAT Traversal	Extra care if via UDP	Fine

Table 3 SIMPLE vs. our method

6. CONCLUSION

In this paper, we have studied the open and XML-based protocol, XMPP and the legacy one, SMPP. By introducing a gateway to bridge XMPP and SMPP, as shown in Figure 8, desktop, notebook, PDA, and mobile handset users may join together to enjoy the instant messaging service with each other, as shown in Figure 13. In addition, as shown in Table 2, the proposed method benefits operators and ICPs significantly. Moreover, as shown in Table 3, it is clear that our proposed method should prefer XMPP to SIMPLE. XMPP and SIMPLE are considered competitive protocols to each other. The SIMPLE work group has made effort to extend for instant messaging service. Similarly, the XMPP community and the XMPP Standards Foundation have also defined a lot of extension for audio and video call services. Despite of these efforts, SIP still remains the protocol of choice for telephony-like services, and XMPP remains for instant messaging and presence services. Although the XMPP is more suitable and flexible for business application, there are still certain works to do for enterprise applications, such as the guaranteed quality of service, mission-critical or time-sensitive message flow support, and service level agreement. They are critical in the design and implementation of the XMPP protocol. Thanks to the extensibility of XML, the above issues can be considered and solved by way of adding new elements to realize desired rules in the future, with backward compatibility to what we have been demonstrated.

REFERENCES

[1] ETSI TS 100 901 V7.4.0 (1999-12), Digital cellular telecommunications system (phase 2+), technical realization of the short message Service (SMS) (GSM 03.40 version 7.4.0 release) 1998.
[2] ETSI TS 100 900 V7.2.0 (1999-07), Digital cellular telecommunications system (phase 2+), alphabets and language-specific information (GSM 03.38 version 7.2.0 release) 1998.
[3] SMPP Developers Forum, http://smsforum.net/doc/public/Spec/SMPP_v3_4_Issue1_2.pdf, Oct. 12, 1999.
[4] P. Saint-Andre, ed., Extensible Messaging and Presence Protocol (XMPP): Core, IETF RFC 3920, Oct. 2004
[5] P. Saint-Andre, ed., Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence, IETF RFC 3921, Oct. 2004
[6] P. Saint-Andre, ed., Mapping the Extensible Messaging and Presence Protocol (XMPP) to Common Presence and Instant Messaging (CPIM), IETF RFC 3922, Oct. 2004
[7] P. Saint-Andre, ed., End-to-End Signing and Object Encryption for the Extensible Messaging and Presence Protocol (XMPP), IETF RFC 3923, Oct. 2004
[8] Peter Saint-Andre and Ralph Meijer, “Streaming XML with Jabber/XMPP,” IEEE Internet Computing, vol. 9, No. 5, pp. 82-89, Sep-Oct. 2005
[9] W3c XML standards, http://www.w3.org/XML/.
[10] Tim Dierks, and et al., "The TLS Protocol Version 1.0", RFC 2246, Jan. 1999.
[11] John G. Myers, "Simple Authentication and Security Layer (SASL)", RFC 2222, Oct. 1997.
[12] The Unicode Standard - http://www.unicode.org.
[13] Tim Berners-Lee, and et al., “Uniform Resource Identifiers (URI): Generic Syntax,” Internet RFC 2396, Aug. 1998.
[14] Roy T. Fielding, and et al., “Hypertext Transfer Protocol -- HTTP/1.1,” Internet RFC 2616, Jun. 1999.
[15] Ben Campbell, and et al., "Session Initiation Protocol (SIP) Extension for Instant Messaging," Internet RFC 3428, Dec. 2002.
[16] http://datatracker.ietf.org/wg/simple/documents/
[17] Hiroyasu Sugano and et al., "Presence Information Data Format (PIDF)," IETF RFC 3863, Aug. 2004
[18] Daniel Senie, "Network Address Translator (NAT)-Friendly Application Design Guidelines," IETF RFC3235, Jan. 2002

Authors

Heng-Te Chu is a lecturer in the Department of Information Networking Technology, Hsiuping University of Science and Technology, Taichung, Taiwan. His research interests cover location-aware management, mobile computing, wireless networking, and SIP-based applications.

Lili Hsieh is an assistant professor in the Department of Information Management, Hsiuping University of Science and Technology, Taichung, Taiwan. His research interests are network commerce, mobile wireless network, and image processing.

Wen-Shiung Chen received the M.S. degree from National Taiwan University, Taipei, Taiwan, in 1985 and the Ph.D. degree from the University of Southern California, Los Angeles, CA, in 1994, both in electrical engineering. He has been with the Department of Electrical Engineering at Feng Chia University, Taichung, Taiwan from 1994 to 2000. In 2000, he joined the Department of Electrical Engineering at National Chi Nan University, where he currently is a Professor. His research interests include digital image processing, image analysis and pattern recognition, computer vision, machine learning, image and video compression, biometrics, mobile computing and networking.