Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 18, 2013

P. Saint-Andre
Cisco Systems, Inc.
E. Gavita
N. Hossain
S. Loreto
Ericsson

October 15, 2012

Interworking between the Session Initiation Protocol (SIP) and the Extensible Messaging and Presence Protocol (XMPP): One-to-One Text Chat

draft-saintandre-sip-xmpp-chat-04

Abstract

This document defines a bi-directional protocol mapping for the exchange of instant messages in the context of a one-to-one chat session between a user of the Session Initiation Protocol (SIP) and a user of the Extensible Messaging and Presence Protocol (XMPP). Specifically for SIP text chat, this document specifies a mapping to the Message Session Relay Protocol (MSRP).

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 18, 2013.

Table of Contents

1. Introduction

1.1. Overview

Both the Session Initiation Protocol [RFC3261] and the Extensible Messaging and Presence Protocol [RFC6120] can be used for the purpose of one-to-one text chat over the Internet. To ensure interworking between these technologies, it is important to define bi-directional protocol mappings.

The architectural assumptions underlying such protocol mappings are provided in [I-D.saintandre-sip-xmpp-core], including mapping of addresses and error conditions. Mappings for single instant messages (sometimes called "pager-mode" messaging) are provided in [I-D.saintandre-sip-xmpp-im]. This document specifies mappings for one-to-one text chat sessions (sometimes called "session-mode" messaging); in particular, this document specifies mappings between XMPP and the Message Session Relay Protocol [RFC4975]. Mapping of multi-user text chat (sometimes called "groupchat") is out of scope for this document.

1.2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

1.3. Scope

Both XMPP and SIP/SIMPLE technologies enable end users to send "instant messages" to other end users. The term "instant message" usually refers to messages sent between two end users for delivery in close to real time (rather than messages that are stored and forwarded to the intended recipient upon request). Generally, there are three kinds of instant messages:

1. Single messages, which are sent from the sender to the recipient outside the context of any one-to-one chat session or multi-user text conference. The message is immediately delivered and not stored in an inbox. In XMPP a single message is a <message/> stanza of type "normal" as specified in [RFC6121]. In SIP/SIMPLE a single message is sent via the MESSAGE method as specified in [RFC3428].
2. One-to-one chat messages, which are sent from the sender to the recipient (i.e., one-to-one) in the context of a "chat session" between the two entities. In XMPP a chat message is a <message/> stanza of type "chat". In SIP/SIMPLE a chat message is sent

using an MSRP session as specified in [RFC4975].

3. Groupchat messages, which are sent from a sender to multiple recipients (i.e., two or more) in the context of a "multi-user chat session", "text conference", or "chatroom". In XMPP a groupchat message is a <message/> stanza of type "groupchat" that is reflected from the sender to multiple recipients by a multi-user chat service, as defined in [XEP-0045]. In SIP/SIMPLE a groupchat message is reflected from the sender to multiple recipients by a conference server that uses MSRP to handle groupchat sessions, as defined in [MSRP-MULTI].

This document covers only scenario #2 for converting XMPP messages of type "chat" to and from their corresponding SIP INVITE and MSRP message types on the SIP/SIMPLE side.

As in [I-D.saintandre-sip-xmpp-im] and related documents, the approach taken here is to directly map syntax and semantics from one protocol to another. The mapping described herein depends on the protocols defined in the following specifications:

- XMPP chat sessions using message stanzas of type "chat" are specified in [RFC6121].
- A method for formally negotiating an XMPP chat session is specified in the Stanza Session Negotiation extension to XMPP [XEP-0155].
- SIP-based chat sessions using the SIP INVITE and SEND request types are specified in [RFC4975].

1.4. Formal and Informal Sessions

The traditional model for a one-to-one chat "session" in XMPP is for a user to simply send a message of type "chat" to a contact, without any formal negotiation of session parameters; the contact would then reply to the message, and the sum total of such messages exchanged during a defined period of time can be considered a chat session. This informal approach to chat sessions in XMPP can be mapped both to SIP pager-mode messaging using the SIP MESSAGE method (as documented in [I-D.saintandre-sip-xmpp-core]) and to an MSRP chat session. How a gateway chooses to map the XMPP chat session to the SIP side is a matter of the implementation, although guidelines are provided under Section 1.5.

However, in XMPP it is also possible to formally request a chat session and negotiate its parameters (e.g., security, privacy, message logging) before beginning the session and exchanging messages. The protocol for doing so is defined in [XEP-0155]. In this case, the XMPP chat session SHOULD be translated into an MSRP session.

This document covers the mapping of both informal and formally-negotiated XMPP chat sessions into MSRP sessions, and from MSRP sessions into XMPP informal and formal sessions.

1.5. Gateway Heuristics

When a gateway receives a chat message or chat session request intended for a recipient that is registered with the gateway itself or has an account on a local service, it SHOULD adhere to the following process in determining whether to (1) initiate a formal chat session with the recipient, (2) initiate an informal chat session with the recipient, or (3) return an error to the sender.

1. If the gateway has knowledge of the recipient's online endpoints (available resources in XMPP or registered UAs in SIP), then it SHOULD discover the capabilities of those endpoints.
2. If the gateway determines that one or more of the endpoints supports formal chat sessions, it SHOULD initiate a formal chat session with one of those endpoints (deciding among the endpoints based on presence information or communications priority).
3. If the gateway determines that none of the endpoints supports formal chat sessions, it SHOULD initiate an informal chat session with one of those endpoints (deciding among the endpoints based on presence information or communications priority).
4. If the gateway does not know if the recipient has any online endpoints, it SHOULD return an appropriate error to the sender.

The methods by which a gateway determines support for various capabilities are protocol-specific. For XMPP a gateway SHOULD use the Service Discovery extension defined in [XEP-0030] or, if it receives presence information from the XMPP endpoint, use the Entity Capabilities extension defined in [XEP-0115]. For MSRP a gateway SHOULD use the Session Description Protocol defined in [RFC4566] in conjunction with a high-level protocol that provides a capability query, such as the SIP OPTIONS request defined in [RFC3261].

1.6. Connection Maintenance

XMPP makes use of long-lived TCP connections. If mobility affecting Layer 3 causes a dropped connection, the connection must be re-established. If mobility does not preserve the IP address, the TCP connection will be dropped. Any TLS session and SASL associations must be re-established if the TCP connection is dropped. XMPP binds directly to TCP in the core specification, so the TCP session must remain open for the entire duration of the chat session. The XMPP Standards Foundation does define protocol extensions enabling transport of XMPP traffic over HTTP (refer to [XEP-0124] and [XEP-0206]), so that individual messages are carried using HTTP and

are more robust in environments such as mobile networks, allowing for better recovery if a TCP session is broken.

SIMPLE is similar when using MSRP. The Message Session Relay Protocol [RFC4975] is a protocol for transmitting instant messages (IM) in the context of a session. The protocol specification describes how the session can be negotiated and established with an offer or answer (see [RFC3264]) using the Session Description Protocol [RFC4566]. In SIMPLE, this exchange is carried using SIP as the signaling protocol. After the TCP connection is established, if it fails for any reason, then an MSRP endpoint MAY choose to re-create such a session using a new SDP exchange in a SIP re-INVITE. SIMPLE also uses the MESSAGE request type for transporting instant messaging outside the context of a session. The MESSAGE request is sent inside the signaling path without establishing any dedicated connection.

1.7. Acknowledgements

Some text in this document was borrowed from [I-D.saintandre-sip-xmpp-core] and from [XEP-0155].

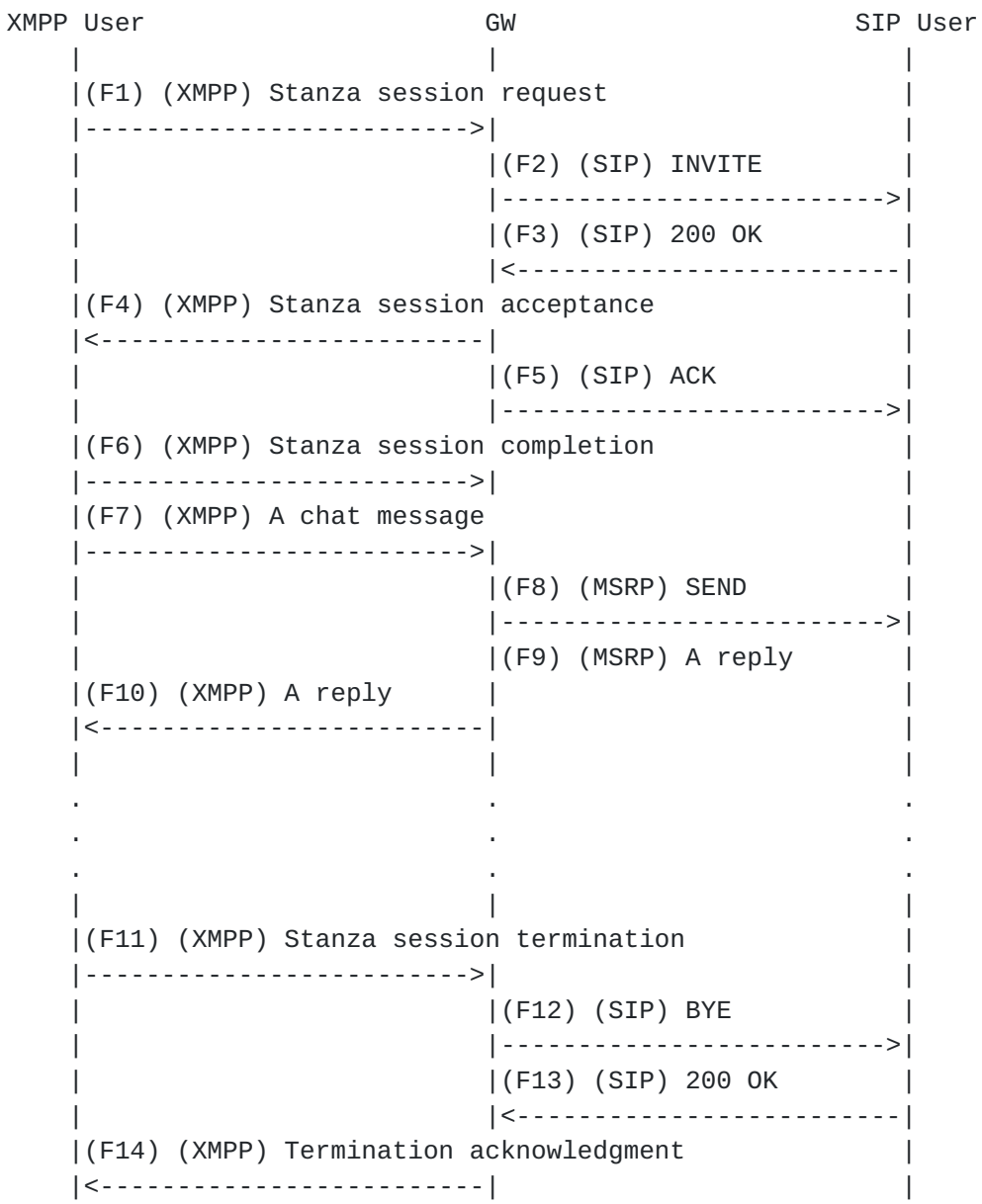
1.8. Discussion Venue

The authors welcome discussion and comments related to the topics presented in this document. The preferred forum is the <sip-xmpp@xmpp.org> mailing list, for which archives and subscription information are available at <http://mail.jabber.org/mailman/listinfo/sip-xmpp>.

2. XMPP Formal Chat Session to MSRP

This section describes how to map an XMPP "formal session" to an MSRP session.

The XMPP formal session is based on the protocol described in [XEP-0155], which enables the initiation, renegotiation, and termination of a formal chat session on the XMPP side. This approach maps to the semantic of the SIP INVITE and BYE methods.



2.1. Initiating a Formal Session

When the XMPP user ("Juliet") wants to initiate a negotiated session with a SIP user ("Romeo"), she sends a <message/> stanza to Romeo containing a <feature/> child qualified by the 'http://jabber.org/protocol/feature-neg' namespace. The <message/> stanza must not contain a <body/> child (as specified in [RFC6121]), since that child element is used for human-readable text. The <message/> stanza type should be "normal". The stanza MUST contain a <thread/> element for tracking purposes (where the newly-generated ThreadID is unique to the proposed session). The encapsulated data form MUST contain a FORM_TYPE field whose type is "hidden" and whose value is "urn:xmpp:ssn"; it must also contain a boolean field named

"accept".

The XMPP user may request a session with a specific resource of the contact. However in this document the resource identifier will be ignored and discarded for cross-system interworking.

Example: (F1) Juliet starts a formal session

```
<message from='juliet@example.com'
         to='romeo@example.net'
         type='normal'>
  <thread>711609sa</thread>
  <feature xmlns='http://jabber.org/protocol/feature-neg'>
    <x xmlns='jabber:x:data' type='form'>
      <title>Open chat with Juliet?</title>
      <field var='FORM_TYPE' type='hidden'>
        <value>urn:xmpp:ssn</value>
      </field>
      <field label='Accept this session?'
             type='boolean'
             var='accept'>
        <value>true</value>
        <required/>
      </field>
      <field label='Primary written language of the chat'
             type='list-single'
             var='language'>
        <value>en</value>
        <option label='English'><value>en</value></option>
        <option label='Italiano'><value>it</value></option>
     </field>
     <field label='XHTML formatting'
            type='list-single'
            var='http://jabber.org/protocol/xhtml-im'>
        <value>may</value>
        <option label='Allow XHTML formatting'>
          <value>may</value>
        </option>
        <option label='Disallow XHTML formatting'>
          <value>mustnot</value>
        </option>
      </field>
    </x>
  </feature>
</message>
```

Upon receiving such a session request, the XMPP server to which Juliet has authenticated attempts to deliver the request to a local

user or attempts to route the request to the remote domain that services the hostname in the 'to' attribute. Naturally, in this document we assume that the hostname in the 'to' attribute is an IM-aware SIP service hosted by a separate server.

As specified in [RFC6121], the XMPP server needs to determine the identity of the remote domain, which it does by performing one or more [RFC2782] lookups. For message stanzas, the order of lookups recommended by [RFC6121] is to first try the "_xmpp-server" service as specified in [RFC6120] and to then try the "_im" service as specified in [RFC3861]. Here we assume that the first lookup will fail but that the second lookup will succeed and return a resolution "_im._simple.example.net.", since we have already assumed that the example.net hostname is running a SIP instant messaging service. (Note: The XMPP server may have previously determined that the remote domain is a SIMPLE server, in which case it would not need to perform the SRV lookups; the caching of such information is a matter of implementation and local service policy, and is therefore out of scope for this document.)

Once the XMPP server (example.com) has determined that the remote domain is serviced by a SIMPLE server, it hands the XMPP message off to its local XMPP-to-SIP gateway (x2s.example.com), which transforms the message into SIP syntax and routes it to the remote SIMPLE server (example.net).

Example: (F2) Juliet starts a formal session (SIP transformation)

```
INVITE sip:romeo@example.net SIP/2.0
To: <sip:romeo@example.net>
From: <sip:juliet@example.com>;tag=786
Subject: Open chat with Juliet?
Call-ID: 711609sa
Content-Type: application/sdp

c=IN IP4 x2s.example.com
m=message 7654 TCP/MSRP *
a=accept-types:text/plain
a=lang:en
a=lang:it
a=path:msrp://x2s.example.com:7654/jshA7weztas;tcp
```

Here the Session Description Protocol offer specifies the MSRP-aware XMPP-to-SIP gateway on the XMPP side as well as other particulars of the session.

There is no direct mapping for the MSRP URIs. In fact MSRP URIs identify a session of instant messages at a particular device; they are ephemeral and have no meaning outside the scope of that session. The authority component of the MSRP URI MUST contain the XMPP-to-SIP gateway hostname or numeric IP address and an explicit port number.

Native XMPP messages as described in [RFC6121] supports text (i.e., UTF-8) only. However, there exists an XMPP extension for XHTML-formatted messges, as defined by the XHTML-IM integration set specified in [XEP-0071]. Unless the use of XHTML-formatted messages is supported by the endpoints or negotiated during session establishment, the "accept-types" attribute that follows an MSRP media line SHOULD indicate "text/plain" as the only media-type that is acceptable to the endpoint; if XHTML is supported or negotiated, the "accept-types" attribute MAY also indicate a media-type of "text/html". (Note: The XHTML-IM integration set supports only a subset of XHTML formatting; it is the responsibility of a gateway to map between full XHTML and XHTML-IM.)

As specified in [I-D.saintandre-sip-xmpp-core], the mapping of XMPP syntax elements to SIP and SDP syntax elements SHOULD be as shown in the following table. (Mappings for elements not mentioned are undefined.)

Table 1: Message syntax mapping from XMPP to SIP/SDP

XMPP Element or Attribute	SIP Header or SDP Contents
<thread/>	Call-ID
from	From
to	To
<title/>	Subject
xml:lang	a=lang:<language tag>
-	a=accept-types:text/plain

2.2. Accepting a Formal Session

Here we assume that the SIP user agent that receives the SIP invitation (containing an offered session description that includes a session of MSRP) accepts the invitation and includes an answer session description that acknowledges the choice of media.

Example: (F3) Romeo accepts the request

```
SIP/2.0 200 OK
To: <sip:romeo@example.net>;tag=087js
From: <sip:juliet@example.com>;tag=786
Call-ID: 711609sa
Content-Type: application/sdp

c=IN IP4 example.net
m=message 12763 TCP/MSRP *
a=accept-types:text/plain
a=lang:it
a=path:msrp://s2x.example.net:12763/kjhd37s2s20w2a;tcp
```

Upon receiving such a response, the SIMPLE server or associated SIP-to-XMPP gateway SHOULD remember that this is a response to a SIP transaction related to an XMPP-SIP translation, based on the SIP Call-ID (which is functionally equivalent to the XMPP <thread/>). The SIP-to-XMPP gateway is responsible for translating the response into an XMPP message stanza and routing it from the SIP user to the XMPP server or associated XMPP-to-SIP gateway.

The SIP-to-XMPP gateway MUST include in the response translation values for all the fields that the XMPP request indicated are required.

Example: (F4) Romeo accepts the request (XMPP translation)

```
<message from='romeo@example.net'
         to='juliet@example.com'
         type='normal'>
  <thread>711609sa</thread>
  <feature xmlns='http://jabber.org/protocol/feature-neg'>
    <x xmlns='jabber:x:data' type='submit'>
      <field var='FORM_TYPE'>
        <value>urn:xmpp:ssn</value>
      </field>
      <field var='accept'><value>true</value></field>
      <field var='language'><value>it</value></field>
    </x>
  </feature>
</message>
```

The SIP-to-XMPP gateway MUST also send a SIP ACK to the SIP user.

Example: (F5) Gateway sends ACK to Romeo's UA

```
ACK sip:romeo@example.net SIP/2.0
To: <sip:romeo@example.net>;tag=087js
From: <sip:juliet@example.com>;tag=786
Call-ID: 711609sa
```

If Romeo accepted the session, Juliet MUST either complete or cancel the stanza session negotiation. The user's client SHOULD verify that the selected values of the fields are acceptable before completing the stanza session negotiation -- and confirming that the session is open -- by replying with the form 'type' attribute set to 'result'. The form MUST contain the FORM_TYPE field and the "accept" field set to "1" or "true".

Example: (F6) Juliet completes negotiation

```
<message from='juliet@example.com'
         to='romeo@example.net'
         type='normal'>
  <thread>711609sa</thread>
  <feature xmlns='http://jabber.org/protocol/feature-neg'>
    <x xmlns='jabber:x:data' type='result'>
      <field var='FORM_TYPE'>
        <value>urn:xmpp:ssn</value>
      </field>
      <field var='accept'><value>true</value></field>
    </x>
  </feature>
</message>
```

Upon receiving such a stanza completing the session negotiation, the XMPP server MUST NOT send any confirmation to the SIP side; instead, it MUST route the acceptance to the SIMPLE server or associated SIP-to-XMPP gateway.

The session is now open and the parties can proceed to exchanging messages.

2.3. Exchanging Messages

Once the session is created, the endpoints can exchange an unbounded number of messages.

The XMPP 'id' attribute is not required in the protocol and there is no way to enforce its use for messages. It is RECOMMENDED to include it as a negotiable item in the SSN negotiation, via the "message-ids" field. However, it is possible that the 'id' will not be present

within the <message/> stanza; in this case the XMPP-to-SIP gateway MUST generate a new unique Message-ID.

If the XMPP user has not explicitly requested message receipts during the negotiation, it is RECOMMENDED that the SIP-to-XMPP gateway shall insert a Failure-Report header field value of "no" during the creation of a SEND request. The XMPP user can include a request for message receipts using the Message Receipts XMPP protocol extension [XEP-0184]; use of this extension can be negotiated via the "urn:xmpp:receipts" field during SSN negotiation.

The mapping of XMPP syntax elements to MSRP syntax elements SHOULD be as shown in the following table. (Mappings for elements not mentioned are undefined.)

Table 2: Message syntax mapping from XMPP Message to MSRP

XMPP Element or Attribute	MSRP Header
to	To-Path
from	From-Path
<body/>	body of the SEND request
-	Content-Type: text/plain
id	Message-ID

The following examples show an exchange of messages.

Example: (F7) Juliet sends an XMPP message

```
<message from='juliet@example.com'
         to='romeo@example.net'
         type='chat'>
  <thread>711609sa</thread>
  <body>Art thou not Romeo, and a Montague?</body>
</message>
```

Example: (F8) Gateway transforms XMPP message to MSRP

```
MSRP a786hjs2 SEND
To-Path: msrp://s2x.example.net:12763/kjhd37s2s20w2a;tcp
From-Path: msrp://x2s.example.com:7654/jshA7weztas;tcp
Message-ID: 87652491
Byte-Range: 1-25/25
Content-Type: text/plain

Art thou not Romeo, and a Montague?
-------a786hjs2$
```

Upon receiving the SEND request, if the request either contains a Failure-Report header field value of "yes" or does not contain a Failure-Report header at all, Romeo's client MUST immediately generate and send a response.

```
MSRP d93kswow 200 OK
To-Path: msrp://x2s.example.com:7654/jshA7weztas;tcp
From-Path: msrp://s2x.example.net:12763/kjhd37s2s20w2a;tcp
-------d93kswow$
```

Romeo can then send a reply using his MSRP user agent.

Example: (F9) Romeo sends a reply

```
MSRP a786hjs2 SEND
To-Path: msrp://x2s.example.com:7654/jshA7weztas;tcp
From-Path: msrp://s2x.example.net:12763/kjhd37s2s20w2a;tcp
Message-ID: 87652491
Byte-Range: 1-25/25
Content-Type: text/plain

Neither, fair saint, if either thee dislike.
-------a786hjs2$
```

The SIP-to-XMPP gateway would then transform that message into appropriate XMPP syntax for routing to the intended recipient.

Example: (F10) Gateway transforms MSRP message to XMPP

```
<message from='romeo@example.net'
         to='juliet@example.com'
         type='chat'>
  <thread>711609sa</thread>
  <body>Neither, fair saint, if either thee dislike.</body>
</message>
```

Note: The size of the XML character data of an XMPP message is not limited by the protocol, but is sometimes limited in deployment. However messages sent using MSRP can be delivered in several SEND requests, so when the XMPP-to-SIP gateway receives a message longer then 2048, it is STRONGLY RECOMMENDED it delivers this message using as few chunks (at least 2048 octets long) as possible.

2.4. Terminating a Formal Session

If Juliet decides to terminate the negotiated chat session, her client sends a <message/> stanza to Romeo containing a data form of type "submit". The <message/> stanza MUST contain a <thread/> element with the same XML character data as the original initiation request. The data form containing a boolean field named "terminate" set to a value of "1" or "true".

Example: (F11) Juliet terminates the chat session

```
<message from='juliet@example.com'
         to='romeo@example.net'
         type='normal'>
  <thread>711609sa</thread>
  <feature xmlns='http://jabber.org/protocol/feature-neg'>
    <x xmlns='jabber:x:data' type='submit'>
      <field var='FORM_TYPE'>
        <value>urn:xmpp:ssn</value>
      </field>
      <field var='terminate'>
        <value>1</value>
      </field>
    </x>
  </feature>
</message>
```

Upon receiving such stanza terminating the chat session, the XMPP-to-SIP gateway terminates the SIP session by sending a SIP BYE to tear down the MSRP session with Romeo's client. Romeo's SIP client then responds with a 200 OK.

Example: (F12) Juliet terminates the chat session (SIP translation)

```
BYE romeo@example.net sip: SIP/2.0
Max-Forwards: 70
From: <sip:juliet@example.com>;tag=786
To: <sip:romeo@example.net>;tag=087js
Call-ID: 711609sa
Cseq: 1 BYE
Content-Length: 0
```

Example: (F13) Romeo acknowledges termination

```
SIP/2.0 200 OK
From: <sip:juliet@example.com>;tag=786
To: <sip:romeo@example.net>;tag=087js
Call-ID: 711609sa
CSeq: 1 BYE
Content-Length: 0
```

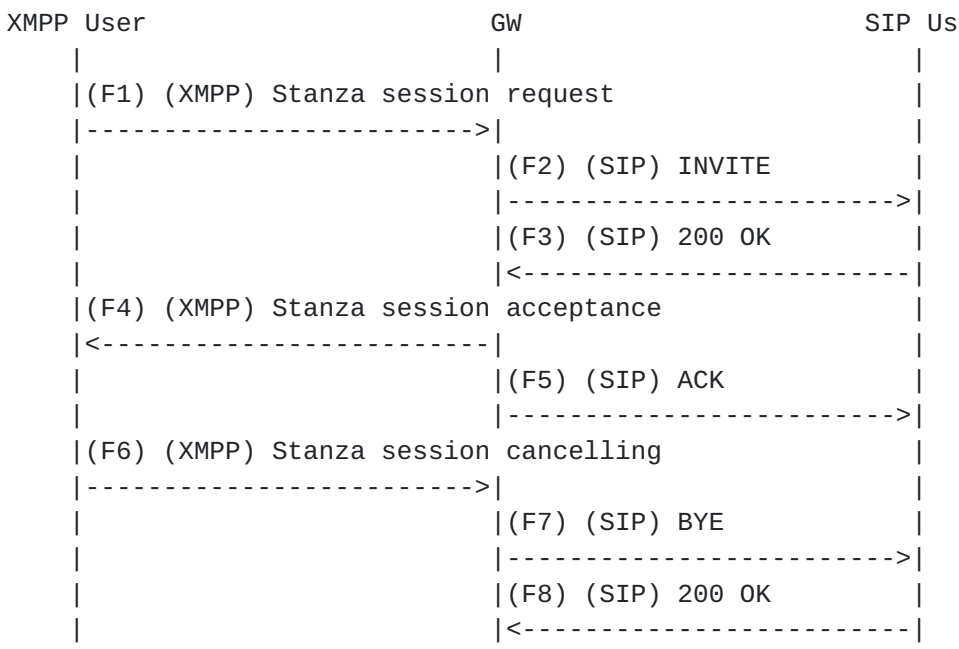
Upon receiving the 200 OK, the SIP-to-XMPP gateway acknowledges the termination of the chat session on the XMPP side by sending a <message/> containing a data form of type "result", and the value of the "terminate" field set to "1" or "true". The client must mirror the <thread/> value it received.

Example: (F14) Romeo acknowledges termination (XMPP translation)

```
<message from='romeo@example.net'
         to='juliet@example.com'
         type='normal'>
  <thread>711609sa</thread>
  <feature xmlns='http://jabber.org/protocol/feature-neg'>
    <x xmlns='jabber:x:data' type='result'>
      <field var='FORM_TYPE'>
        <value>urn:xmpp:ssn</value>
      </field>
      <field var='terminate'>
        <value>1</value>
      </field>
    </x>
  </feature>
</message>
```

2.5. Cancelling the Negotiation

If Romeo accepted the session but Juliet decides to cancel the stanza session negotiation, the flow is as follows.



That is, Juliet's client shall reply with a data form containing the FORM_TYPE field and the "accept" field set to "0" or "false":

Example: (F6) User cancels stanza session negotiation

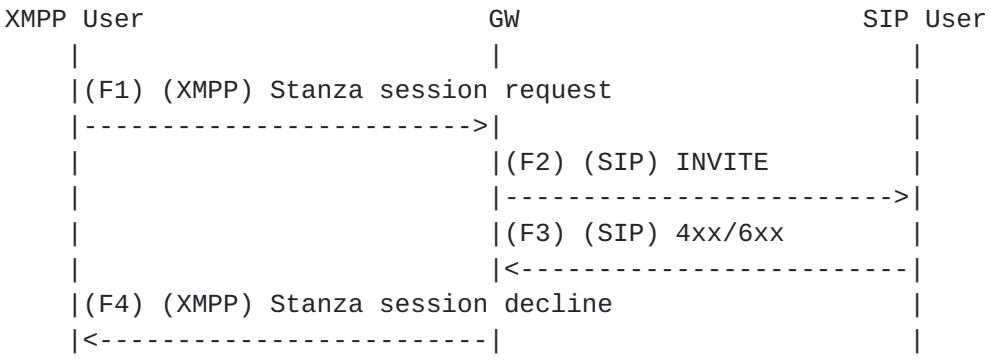
```
<message from='juliet@example.com'
         to='romeo@example.net'
         type='normal'>
  <thread>711609sa</thread>
  <feature xmlns='http://jabber.org/protocol/feature-neg'>
    <x xmlns='jabber:x:data' type='result'>
      <field var='FORM_TYPE'>
        <value>urn:xmpp:ssn</value>
      </field>
      <field var='accept'>
        <value>0</value>
      </field>
    </x>
  </feature>
</message>
```

Upon receiving such stanza cancelling the session negotiation, the XMPP-to-SIP gateway MUST send a SIP BYE. Once the XMPP-to-SIP gateway receives the 200 OK, the internal session data is removed and the session is officially cancelled also in the SIP-to-XMPP gateway.

If the SIP user had sent any messages to XMPP while awaiting confirmation of the session, the SIP-to-XMPP gateway MUST return them to the SIP user with an appropriate error.

2.6. Rejecting a Formal Session

A common scenario occurs when the SIP UA is currently unwilling or unable to accept a formal session, in which case the flow is as follows.



Here the SIP UA declining an offer contained in an INVITE SHOULD return a 4xx or a 6xx response, such as 406 Not Acceptable or 603 Decline. Such a response SHOULD include a Warning header field value explaining why the offer was rejected.

Example: (F3) SIP user declines offer and specifies reason (SIP)

```
SIP/2.0 603 Decline
From: <sip:juliet@example.com>;tag=786
To: <sip:romeo@example.net>;tag=087js
Call-ID: 711609sa
Warning: I'm busy!
Content-Length: 0
```

Upon receiving the error response for the SIP INVITE, the XMPP-to-SIP gateway shall send a "Session Reject" message back to the XMPP Client. This message contains a data form that MUST contain the FORM_TYPE field and the "accept" field set to "0" or "false". It is RECOMENDED that the form does not contain any other field even if the request indicated they are required. The client MAY include a reason in the <body/> child of the <message/> stanza. The content of the Warning header field present in the SIP response SHOULD be mapped to a "reason" field in the data form. If the Warning header is not present then the descriptive phrase of the SIP response can be used.

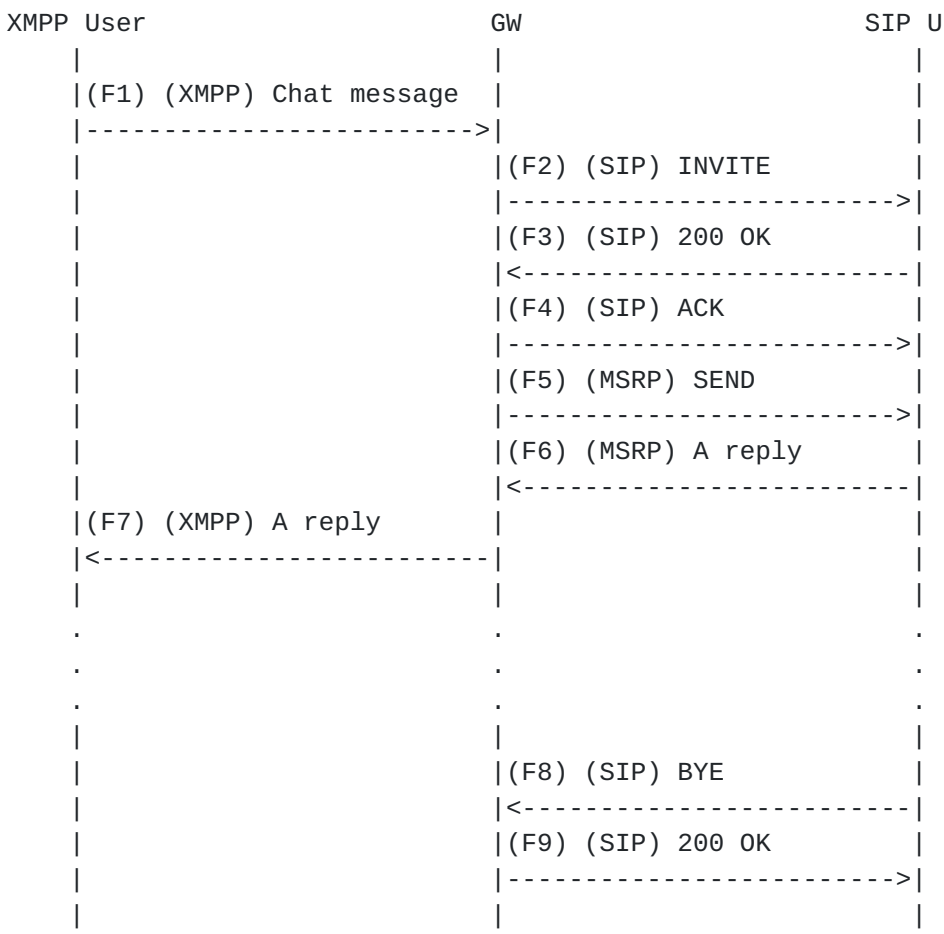
Example: (F4) SIP user declines offer and specifies reason (XMPP translation)

```
<message from='romeo@example.net'
         to='juliet@example.com'
         type='normal'>
  <thread>711609sa</thread>
  <feature xmlns='http://jabber.org/protocol/feature-neg'>
    <x xmlns='jabber:x:data' type='submit'>
      <field var='FORM_TYPE'>
        <value>urn:xmpp:ssn</value>
      </field>
      <field var='accept'>
        <value>0</value>
      </field>
      <field var='reason'>
        <value>I&apos;m busy!</value>
      </field>
    </x>
  </feature>
</message>
```

3. XMPP Informal Session to MSRP

In XMPP, the "informal session" approach is to simply send someone a <message/> of type "chat" without starting any session negotiation ahead of time (as described in [RFC6121]). The XMPP "informal session" approach maps very well into a SIP MESSAGE request, as described in [I-D.saintandre-sip-xmpp-core]. However, the XMPP informal session approach can also be mapped to MSRP if the XMPP-to-SIP gateway maintains additional state.

The order of events is as follows.



First the XMPP user would generate an XMPP chat message.

Example: (F1) Juliet sends an XMPP message

```
<message from='juliet@example.com'
         to='romeo@example.net'
         type='chat'>
  <thread>711609sa</thread>
  <body>Art thou not Romeo, and a Montague?</body>
</message>
```

The local SIP-to-XMPP gateway at the SIMPLE server would then determine if Romeo supports MSRP. If so, the SIP-to-XMPP gateway would initiate an MSRP session with Romeo on Juliet's behalf.

Example: (F2) Gateway starts a formal session on behalf of Juliet

```
INVITE sip:romeo@example.net SIP/2.0
To: <sip:romeo@example.net>
From: <sip:juliet@example.com>;tag=786
Subject: Open chat with Juliet?
Call-ID: 711609sa
Content-Type: application/sdp

c=IN IP4 x2s.example.com
m=message 7654 TCP/MSRP *
a=accept-types:text/plain
a=lang:en
a=lang:it
a=path:msrp://x2s.example.com:7654/jshA7weztas;tcp
```

Here we assume that Romeo accepts the MSRP session request.

Example: (F3) Romeo accepts the request

```
SIP/2.0 200 OK
To: <sip:romeo@example.net>;tag=087js
From: <sip:juliet@example.com>;tag=786
Call-ID: 711609sa
Content-Type: application/sdp

c=IN IP4 s2x.example.net
m=message 12763 TCP/MSRP *
a=accept-types:text/plain
a=lang:it
a=path:msrp://s2x.example.net:12763/kjhd37s2s20w2a;tcp
```

The XMPP-to-SIP gateway then acks the session acceptance on behalf of Juliet.

Example: (F4) Gateway sends ACK to Romeo's UA

```
ACK sip:romeo@example.net SIP/2.0
To: <sip:romeo@example.net>;tag=087js
From: <sip:juliet@example.com>;tag=786
Call-ID: 711609sa
```

The XMPP-to-SIP gateway then transforms the original XMPP chat message into MSRP.

Example: (F5) Gateway transforms XMPP message to MSRP

```
MSRP a786hjs2 SEND
From-Path: msrp://x2s.example.com:7654/jshA7weztas;tcp
To-Path: msrp://s2x.example.net:12763/kjhd37s2s20w2a;tcp
Message-ID: 87652491
Byte-Range: 1-25/25
Content-Type: text/plain

Art thou not Romeo, and a Montague?
-------a786hjs2$
```

Romeo can then send a reply using his MSRP user agent.

Example: (F6) Romeo sends a reply

```
MSRP a786hjs2 SEND
To-Path: msrp://x2s.example.com:7654/jshA7weztas;tcp
From-Path: msrp://s2x.example.net:12763/kjhd37s2s20w2a;tcp
Message-ID: 87652491
Byte-Range: 1-25/25
Failure-Report: no
Content-Type: text/plain

Neither, fair saint, if either thee dislike.
-------a786hjs2$
```

Note: As previously described, if the users have not negotiated the use message receipts, it is RECOMMENDED that the SIP-to-XMPP gateway shall insert a Failure-Report header field value of "no" during the creation of a SEND request.

The SIP-to-XMPP gateway would then transform that message into appropriate XMPP syntax for routing to the intended recipient.

Example: (F7) Gateway transforms MSRP message to XMPP

```
<message from='romeo@example.net'
         to='juliet@example.com'
         type='chat'>
  <thread>711609sa</thread>
  <body>Neither, fair saint, if either thee dislike.</body>
</message>
```

When the MSRP user wishes to end the chat session, the user's MSRP client sends a SIP BYE.

Example: (F8) Romeo terminates the chat session

```
BYE juliet@example.com sip: SIP/2.0
Max-Forwards: 70
From: <sip:romeo@example.net>;tag=087js
To: <sip:juliet@example.com>;tag=786
Call-ID: 711609sa
Cseq: 1 BYE
Content-Length: 0
```

The BYE is then acknowledged by the XMPP-to-SIP gateway.

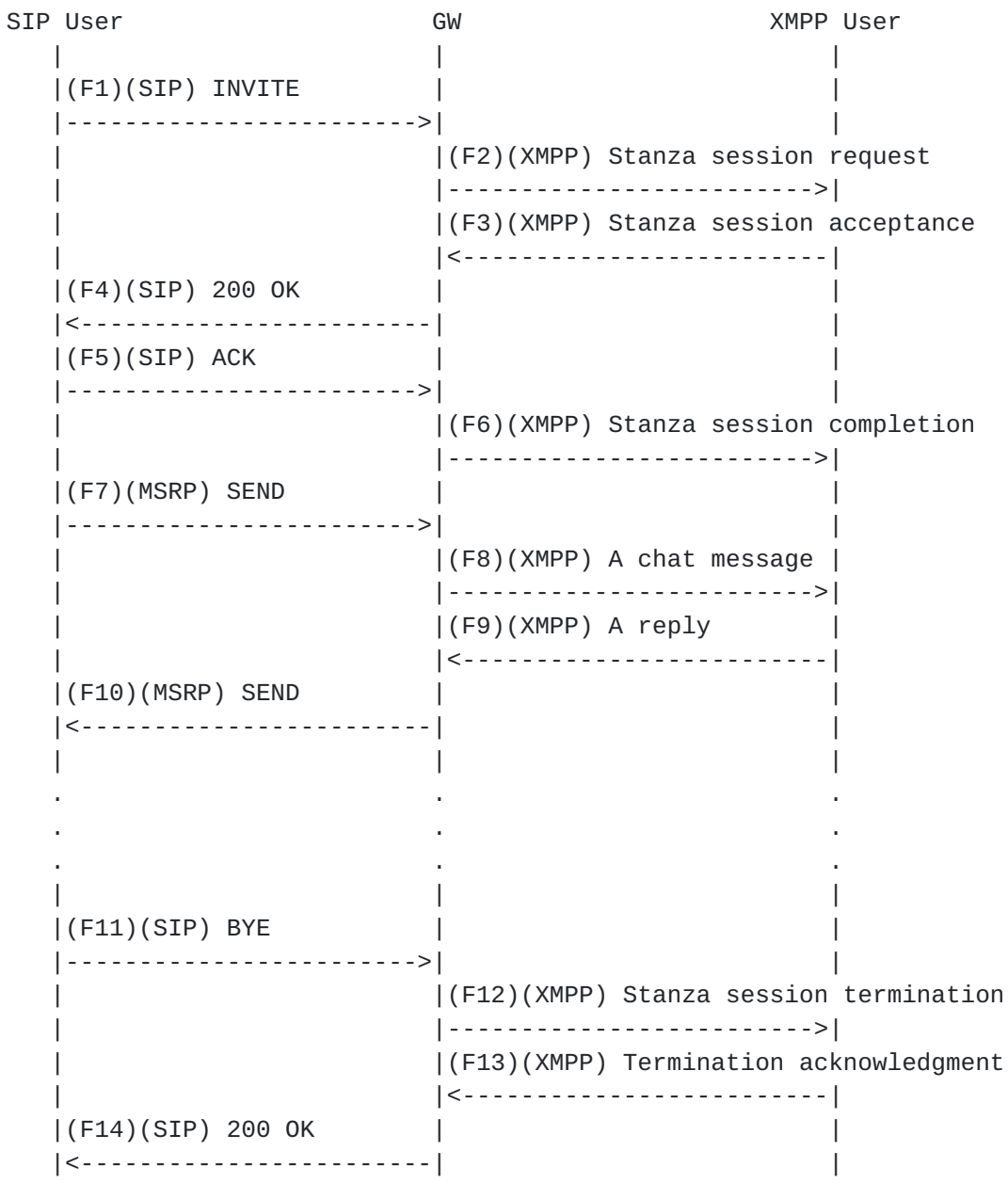
Example: (F9) Gateway acknowledges termination

```
SIP/2.0 200 OK
From: <sip:juliet@example.com>;tag=786
To: <sip:romeo@example.net>;tag=087js
Call-ID: 711609sa
CSeq: 1 BYE
Content-Length: 0
```

4. MSRP to XMPP Formal Session

Unlike the XMPP protocol, the MSRP protocol offers only one way to initiate a chat session, typically using the Session Description Protocol [RFC4566] via the SIP offer/answer mechanism (see [RFC3264]).

The order of events is as follows.



4.1. Initiating a Session

When Romeo wants to start an MSRP message session with Juliet, he first has to start the SIP session by sending out a SIP INVITE request containing an offered session description that includes an MSRP media line accompanied by a mandatory "path" attribute and corresponding URIs. The MSRP media line is also accompanied by an "accept-types" attribute used to specify the only media-types acceptable for Romeo (i.e., text/plain and/or text/html).

Note: In addition to plain text messages, MSRP is able to carry arbitrary (binary) Multipurpose Internet Mail Extensions [RFC2045]

compliant content, such as images or video clips. Disposition of media types other than text/plain and text/html is out of scope for this specification and is a matter of implementation.

Example: (F1) SIP user starts the session

```
INVITE sip:juliet@example.com SIP/2.0
To: <sip:juliet@example.com>
From: <sip:romeo@example.net>;tag=576
Subject: Open chat with Romeo?
Call-ID: 742507no
Content-Type: application/sdp

c=IN IP4 s2x.example.net
m=message 7313 TCP/MSRP *
a=accept-types:text/plain
a=lang:en
a=lang:it
a=path:msrp://s2x.example.net:7313/ansp71weztas;tcp
```

Upon receiving the INVITE, the SIP-to-XMPP gateway needs to determine the identity of the remote domain, which it does by performing one or more DNS SRV lookups [RFC2782]. The SIP-to-XMPP gateway SHOULD resolve the address present in the To header of the INVITE to an im URI, then follow the rules in [RFC3861] regarding the "_im" SRV service for the target domain contained in the To header. If SRV address resolution fails for the "_im" service, the SIP-to-XMPP gateway MAY attempt a lookup for the "_xmpp-server" service as specified in [RFC6120] or MAY return an error to the sender (i.e. 502 Bad Gateway).

If SRV address resolution succeeds, the SIP-to-XMPP gateway is responsible for translating the request into an XMPP message stanza to initiate a negotiated session from the SIP user to the XMPP user.

Example: (F2) SIP user starts the session (XMPP transformation)

```
<message from='romeo@example.net'
         to='juliet@example.com'
         type='normal'>
  <thread>742507no</thread>
  <feature xmlns='http://jabber.org/protocol/feature-neg'>
    <x xmlns='jabber:x:data' type='form'>
      <title>Open chat with Romeo?</title>
      <field var='FORM_TYPE' type='hidden'>
        <value>urn:xmpp:ssn</value>
      </field>
      <field label='Accept this session?' type='form' var='accept'>
        <value>true</value>
        <required/>
      </field>
      <field label='Primary written language of the chat'
             type='list-single'
             var='language'>
        <value>en</value>
        <option label='English'><value>en</value></option>
        <option label='Italiano'><value>it</value></option>
      </field>
    </x>
  </feature>
</message>
```

The mapping of SIP and SDP syntax elements to XMPP syntax elements SHOULD be as shown in the following table. (Mappings for elements not mentioned in the foregoing table are undefined.)

Table 3: Message syntax mapping from SIP to XMPP

SIP Header or SDP Contents	XMPP Element or Attribute
Call-ID	<thread/>
From	from
To	to
Subject	<title/>
accept-types	-
a=lang	xml:lang
To	to

See previous note regarding negotiation and use of the XHTML-IM integration set for XHTML-formatted messages (i.e., the "text/html" accept-type).

4.2. Accepting a Session

If the request is accepted then Juliet's client MUST include all the fields that were marked as required in the request message.

In the example below, we assume that Juliet accepts the session and specifies that she prefers to speak Italian with Romeo.

Example: (F3) Juliet accepts session and specifies parameters

```
<message from='juliet@example.com'
         to='romeo@example.net'
         type='normal'>
  <thread>742507no</thread>
  <feature xmlns='http://jabber.org/protocol/feature-neg'>
    <x xmlns='jabber:x:data' type='submit'>
      <field var='FORM_TYPE'>
        <value>urn:xmpp:ssn</value>
      </field>
      <field var='accept'><value>true</value></field>
      <field var='language'><value>it</value></field>
    </x>
  </feature>
</message>
```

Upon receiving such a response, the SIP-to-XMPP gateway SHOULD remember that this is a response to a stanza related to an SIP-XMPP translation, based on the SIP Call-ID. The SIP-to-XMPP gateway is responsible for translating the response into a SIP response and delivering it from the XMPP user back to the SIP user.

Example: (F4) Juliet accepts session (SIP translation)

```
SIP/2.0 200 OK
To: <sip:juliet@example.com>;tag=534
From: <sip:romeo@example.net>;tag=576
Call-ID: 742507no
Content-Type: application/sdp

c=IN IP4 x2s.example.com
m=message 8763 TCP/MSRP *
a=accept-types:text/plain
a=lang:it
a=path:msrp://x2s.example.com:8763/lkjh37s2s20w2a;tcp
```

4.3. Completing the Transaction

In this case, the 200 OK is routed back and is received by Romeo's UA. Finally, Romeo's client sends an acknowledgment message, ACK, to Juliet's client to confirm the reception of the final response (200 OK).

Example: (F5) Romeo sends ACK

```
ACK sip:juliet@example.com SIP/2.0
To: <sip:juliet@example.com>;tag=534
From: <sip:romeo@example.net>;tag=576
Call-ID: 742507no
```

Upon receiving the ACK, the SIP-to-XMPP gateway SHOULD remember this is an acknowledgment to an XMPP formal session. The SIP-to-XMPP gateway is responsible for translating the acknowledgment into a confirmation stanza, without inserting other content (e.g. a <body/> element cannot be inserted).

Example: (F6) Romeo sends ACK (XMPP translation)

```
<message from='romeo@example.net'
         to='juliet@example.com'
         type='normal'>
  <thread>742507no</thread>
  <feature xmlns='http://jabber.org/protocol/feature-neg'>
    <x xmlns='jabber:x:data' type='result'>
      <field var='FORM_TYPE'>
        <value>urn:xmpp:ssn</value>
      </field>
      <field var='accept'>
        <value>true</value>
      </field>
    </x>
  </feature>
</message>
```

4.4. Exchanging Messages

When Romeo wants to send a message, he creates an MSRP SEND request that contains the message.

Example: (F7) Romeo sends a message

```
MSRP ad49kswow SEND
To-Path: msrp://x2s.example.com:8763/lkjh37s2s20w2a;tcp
From-Path: msrp://s2x.example.net:7313/ansp71weztas;tcp
Message-ID: 44921zaqwsx
Byte-Range: 1-32/32
Failure-Report: no
Content-Type: text/plain

I take thee at thy word ...
-------ad49kswow$
```

Upon receiving the MSRP SEND request, the SIP-to-XMPP gateway SHOULD remember that the message is for an XMPP user. The SIP-to-XMPP gateway is responsible for translating the MSRP SEND request into an XMPP message stanza.

Example: (F8) Romeo sends a message (XMPP translation)

```
<message from='romeo@example.net'
         to='juliet@example.com'
         type='chat'>
  <thread>742507no</thread>
  <body>I take thee at thy word ...</body>
</message>
```

The mapping of MSRP syntax elements to XMPP syntax elements SHOULD be as shown in the following table. (Mappings for elements not mentioned are undefined.)

Table 4: Message syntax mapping from MSRP Message to XMPP

MSRP Header	XMPP Element or Attribute
To-Path	to
From-Path	from
body of the SEND request	<body/>
Content-Type: text/plain	-
Message-ID	id

Upon receiving the chat message, Juliet can send a reply.

Example: (F9) Juliet sends a reply

```
<message from='juliet@example.com'
         to='romeo@example.net'
         type='chat'>
  <thread>711609sa</thread>
  <body>What man art thou ...?</body>
</message>
```

Example: (F10) Gateway transforms XMPP message to MSRP

```
MSRP a786hjs2 SEND
From-Path: msrp://x2s.example.com:8763/lkjh37s2s20w2a;tcp
To-Path: msrp://s2x.example.net:7313/jshA7weztas;tcp
Message-ID: 87652491
Byte-Range: 1-25/25
Failure-Report: no
Content-Type: text/plain

What man art thou ...?
-------a786hjs2$
```

4.5. Terminating a Session

When Romeo wants to terminate the session, he is required to send a SIP BYE request.

Example: (F11) Romeo terminates the session

```
BYE juliet@example.com sip: SIP/2.0
Max-Forwards: 70
From: <sip:romeo@example.net>;tag=576
To: <sip:juliet@example.com>;tag=534
Call-ID: 742507no
Cseq: 1 BYE
Content-Length: 0
```

Upon receiving the SIP BYE request, the XMPP-to-SIP gateway SHOULD translate the request to a <message/> stanza containing a data form of type "submit". The <message/> element MUST contain a <thread/> element with the same XML character data as the original initiation request. The data form containing a boolean field named "terminate" should be set to a value of "1" or "true".

Example: (F12) Romeo terminates the session (XMPP translation)

```
<message from='romeo@example.net'
         to='juliet@example.com'
         type='normal'>
  <thread>742507no</thread>
  <feature xmlns='http://jabber.org/protocol/feature-neg'>
    <x xmlns='jabber:x:data' type='submit'>
      <field var='FORM_TYPE'>
        <value>urn:xmpp:ssn</value>
      </field>
      <field var='terminate'>
        <value>1</value>
      </field>
    </x>
  </feature>
</message>
```

Juliet explicitly acknowledges the termination of the chat session on the XMPP side by sending a <message/> containing a data form of type "result", and the value of the "terminate" field set to "1" or "true". The client MUST mirror the <thread/> value it received.

Example: (F13) Juliet acknowledges the termination of the session

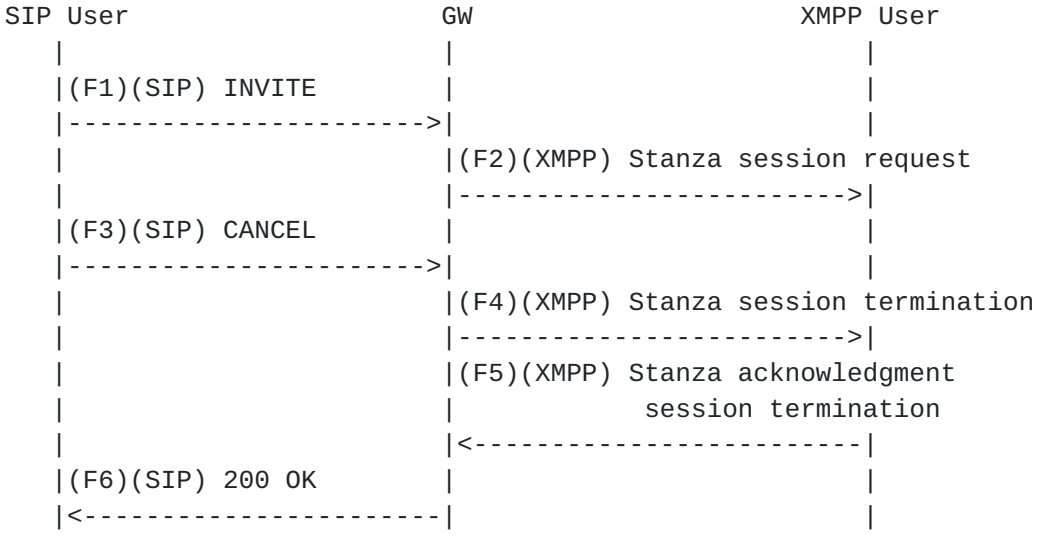
```
<message from='juliet@example.com'
         to='romeo@example.net'
         type='normal'>
  <thread>742507no</thread>
  <feature xmlns='http://jabber.org/protocol/feature-neg'>
    <x xmlns='jabber:x:data' type='result'>
      <field var='FORM_TYPE'>
        <value>urn:xmpp:ssn</value>
      </field>
      <field var='terminate'>
        <value>1</value>
      </field>
    </x>
  </feature>
</message>
```

Upon receiving the acknowledgment message, the XMPP-to-SIP gateway SHOULD translate it to a SIP answer 200 OK.

Example: (F14) Juliet acknowledges the termination of the session (SIP translation)

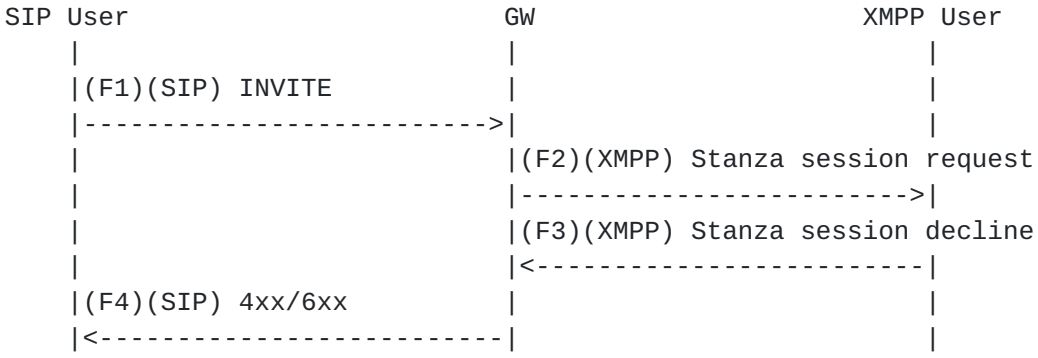
```
SIP/2.0 200 OK
From: <sip:romeo@example.net>;tag=576
To: <sip:juliet@example.com>;tag=534
Call-ID: 742507no
CSeq: 1 BYE
```

4.6. Cancelling the Transaction



A common scenario occurs when the SIP user issues an invitation to set up a chat session with an XMPP user and immediately after the SIP invitation is sent, the SIP user decides to cancel it. The SIP-to-XMPP gateway will receive the CANCEL request and using the Call-ID, To, From and CSeq (sequence number only) header field values as a guide, will issue an XMPP stanza session termination request to the XMPP user to cancel the XMPP formal session (assuming that it was already set up). Once the XMPP-to-SIP gateway receives an ACK stanza message for the session termination, the XMPP-to-SIP gateway will respond with a status of 200 (OK) back to the SIP user. It is important to note that if the SIP session transaction does not exist, the XMPP-to-SIP gateway will return a status of 481 (Transaction Does Not Exist) back to the SIP User.

4.7. Rejecting the Transaction



Another common scenario occurs when the XMPP UA is currently not willing or able to accept a formal session request. The XMPP UA SHOULD decline the invitation. The data form MUST contain the FORM_TYPE field and the "accept" field set to "0" or "false". It is RECOMMENDED that the form does not contain any other fields even if the request indicated they are required.

Example: (F3) User declines offer

```
<message from='juliet@example.com'
         to='romeo@example.net'
         type='normal'>
  <thread>742507no</thread>
  <feature xmlns='http://jabber.org/protocol/feature-neg'>
    <x xmlns='jabber:x:data' type='submit'>
      <field var='FORM_TYPE'>
        <value>urn:xmpp:ssn</value>
      </field>
      <field var='accept'><value>0</value></field>
    </x>
  </feature>
</message>
```

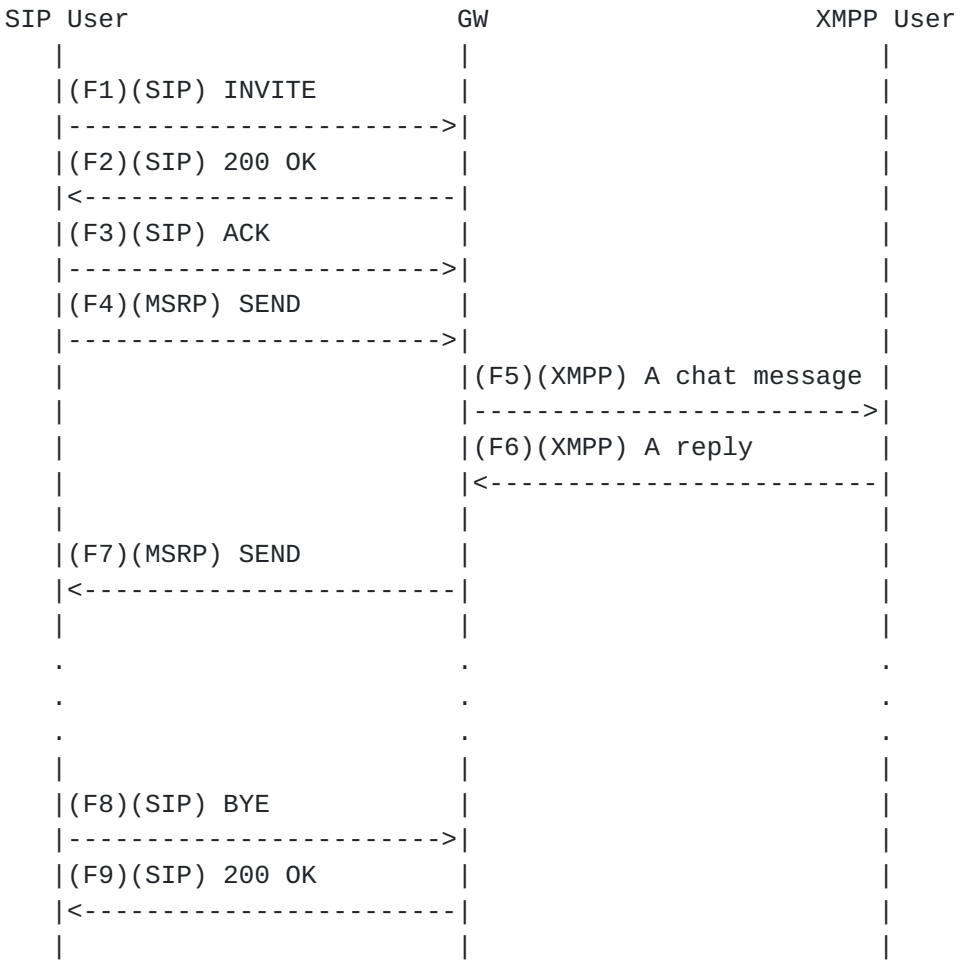
Upon receiving the declined response for the XMPP formal session request, the XMPP-to-SIP gateway SHOULD return a 4xx or a 6xx SIP response back to the SIP client.

4.8. Session Negotiation Fails

If the XMPP recipient of a formal session request does not support stanza session negotiation as specified in [XEP-0155], it will return an XMPP <service-unavailable/> stanza error. Upon receiving this error from the XMPP recipient, the XMPP-to-SIP gateway SHOULD return a 501 SIP response back to the SIP sender.

5. MSRP to XMPP Informal Session

When an MSRP client sends messages through a gateway to an XMPP client that does not support formal sessinos, the order of events is as follows.



Example: (F1) SIP user starts the session

```
INVITE sip:juliet@example.com SIP/2.0
To: <sip:juliet@example.com>
From: <sip:romeo@example.net>;tag=576
Subject: Open chat with Romeo?
Call-ID: 742507no
Content-Type: application/sdp

c=IN IP4 s2x.example.net
m=message 7313 TCP/MSRP *
a=accept-types:text/plain
a=lang:en
a=lang:it
a=path:msrp://s2x.example.net:7313/ansp71weztas;tcp
```

Example: (F2) Gateway accepts session on Juliet's behalf

```
SIP/2.0 200 OK
To: <sip:juliet@example.com>;tag=534
From: <sip:romeo@example.net>;tag=576
Call-ID: 742507no
Content-Type: application/sdp

c=IN IP4 x2s.example.com
m=message 8763 TCP/MSRP *
a=accept-types:text/plain
a=lang:it
a=path:msrp://x2s.example.com:8763/lkjh37s2s20w2a;tcp
```

Example: (F3) Romeo sends ACK

```
ACK sip:juliet@example.com SIP/2.0
To: <sip:juliet@example.com>;tag=534
From: <sip:romeo@example.net>;tag=576
Call-ID: 742507no
```

Example: (F4) Romeo sends a message

```
MSRP ad49kswow SEND
To-Path: msrp://x2s.example.com:8763/lkjh37s2s20w2a;tcp
From-Path: msrp://s2x.example.net:7313/ansp71weztas;tcp
Message-ID: 44921zaqwsx
Byte-Range: 1-32/32
Failure-Report: no
Content-Type: text/plain

I take thee at thy word ...
-------ad49kswow$
```

Example: (F5) Romeo sends a message (XMPP translation)

```
<message from='romeo@example.net'
         to='juliet@example.com'
         type='chat'>
  <thread>742507no</thread>
  <body>I take thee at thy word ...</body>
</message>
```

Example: (F6) Juliet sends a reply

```
<message from='juliet@example.com'
         to='romeo@example.net'
         type='chat'>
  <thread>711609sa</thread>
  <body>What man art thou ...?</body>
</message>
```

Example: (F8) Gateway transforms XMPP message to MSRP

```
MSRP a786hjs2 SEND
To-Path: msrp://s2x.example.net:7313/jshA7weztas;tcp
From-Path: msrp://x2s.example.com:8763/lkjh37s2s20w2a;tcp
Message-ID: 87652491
Byte-Range: 1-25/25
Failure-Report: no
Content-Type: text/plain

What man art thou ...?
-------a786hjs2$
```

Example: (F9) Romeo terminates the session

```
BYE juliet@example.com sip: SIP/2.0
Max-Forwards: 70
From: <sip:romeo@example.net>;tag=576
To: <sip:juliet@example.com>;tag=534
Call-ID: 742507no
Cseq: 1 BYE
Content-Length: 0
```

Example: (F10) Gateway acknowledges the termination of the session on behalf of XMPP user

```
SIP/2.0 200 OK
From: <sip:romeo@example.net>;tag=576
To: <sip:juliet@example.com>;tag=534
Call-ID: 742507no
CSeq: 1 BYE
```

6. Security Considerations

To follow.

7. IANA Considerations

This document requests no actions of IANA.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, June 2002.

[RFC3861] Peterson, J., "Address Resolution for Instant Messaging and Presence", RFC 3861, August 2004.

[RFC4975] Campbell, B., Mahy, R., and C. Jennings, "The Message Session Relay Protocol (MSRP)", RFC 4975, September 2007.

[RFC6120] Saint-Andre, P., "Extensible Messaging and Presence Protocol (XMPP): Core", RFC 6120, March 2011.

[RFC6121] Saint-Andre, P., "Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence", RFC 6121, March 2011.

[XEP-0155] Paterson, I. and P. Saint-Andre, "Stanza Session Negotiation", XSF XEP 0155, January 2008.

8.2. Informative References

[I-D.saintandre-sip-xmpp-core] Saint-Andre, P., Houri, A., and J. Hildebrand, "Interworking between the Session Initiation Protocol (SIP) and the Extensible Messaging and Presence Protocol (XMPP): Core", draft-saintandre-sip-xmpp-core-02 (work in progress), October 2012.

[I-D.saintandre-sip-xmpp-im] Saint-Andre, P., Houri, A., and J. Hildebrand, "Interworking between the Session Initiation Protocol (SIP) and the Extensible Messaging and Presence Protocol (XMPP): Instant Messaging", draft-saintandre-sip-xmpp-im-01 (work in progress), March 2009.

[MSRP-MULTI] Niemi, A., Garcia-Martin, M., and G. Sandbakken, "Multi-party Instant Message (IM) Sessions Using the Message Session Relay Protocol (MSRP)", draft-ietf-simple-chat-16 (work in progress), August 2012.

[RFC2045] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.

[RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, February 2000.

[RFC3428] Campbell, B., Rosenberg, J., Schulzrinne, H., Huitema, C., and D. Gurle, "Session Initiation Protocol (SIP) Extension for Instant Messaging", RFC 3428, December 2002.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, July 2006.

[XEP-0030] Hildebrand, J., Millard, P., Eatmon, R., and P. Saint-Andre, "Service Discovery", XSF XEP 0030, June 2008.

[XEP-0045] Saint-Andre, P., "Multi-User Chat", XSF XEP 0045, April 2007.

[XEP-0071] Saint-Andre, P., "XHTML-IM", XSF XEP 0071, August 2007.

[XEP-0115] Hildebrand, J., Saint-Andre, P., Troncon, R., and J. Konieczny, "Entity Capabilities", XSF XEP 0115, February 2008.

[XEP-0124] Paterson, I., Smith, D., and P. Saint-Andre, "Bidirectional-streams Over Synchronous HTTP (BOSH)", XSF XEP 0124, October 2008.

[XEP-0184] Saint-Andre, P. and J. Hildebrand, "Message Receipts", XSF XEP 0184, September 2007.

[XEP-0206] Paterson, I., "XMPP Over BOSH", XSF XEP 0206, October 2008.

Authors' Addresses

Peter Saint-Andre
Cisco Systems, Inc.
1899 Wynkoop Street, Suite 600
Denver, CO 80202
USA

Phone: +1-303-308-3282
Email: psaintan@cisco.com

Eddy Gavita
Ericsson
Decarie Boulevard
Town of Mount Royal, Quebec
Canada

Email: eddy.gavita@ericsson.com

Nazin Hossain
Ericsson
Decarie Boulevard
Town of Mount Royal, Quebec
Canada

Email: Nazin.Hossain@ericsson.com

Salvatore Loreto
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

Email: Salvatore.Loreto@ericsson.com