SIP D. Worley
Internet-Draft Nortel Networks
Intended status: BCP March 5, 2009
Expires: September 6, 2009

Session Initiation Protocol Service Example -- Music on Hold
draft-worley-service-example-03

Status of this Memo

Copyright Notice

Abstract

The "music on hold" feature is one of the most desired features of telephone systems in the business environment. "Music on hold" is where, when one party to a call has the call "on hold", that party's telephone provides an audio stream (often music) to be heard by the other party. Architectural features of SIP make it difficult to implement music-on-hold in a way that is fully compliant with the standards. The implementation of music-on-hold described in this document is fully effective and standards-compliant, but is simpler than the methods previously documented.

Table of Contents

1. Introduction

Within SIP[sip]-based systems, it is desirable to be able to provide features that are similar to those provided by traditional telephony systems. A frequently requested feature is "music on hold": The music-on-hold feature is where, when one party to a call has the call "on hold", that party's telephone provides an audio stream (often music) to be heard by the other party.

Architectural features of SIP make it difficult to implement music-on-hold in a way that is fully compliant with the standards. The purpose of this document is to describe a method that is reasonably simple yet fully effective and standards-compliant.

2. Technique

The essence of the technique is that when the executing UA (the user's UA) performs a re-INVITE of the remote UA to establish the hold state, it provides no SDP[sdp] offer[offer-answer][offer-answer-bis], thus compelling the remote UA to provide an SDP offer. The executing UA then extracts the offer SDP from the remote UA's 2xx response, and uses that as the offer SDP in a new INVITE to the external media source. The external media source is thus directed to provide media directly to the remote UA. The media source's answer SDP is returned to the remote UA in the ACK to the re-INVITE.

2.1. Placing a Call on Hold and Providing an External Media Stream

1. The executing user instructs the executing UA to put the dialog on-hold.

2. The executing UA sends a re-INVITE without SDP to the remote UA, which forces the remote UA to provide an SDP offer in its 2xx response. The Contact header of the re-INVITE includes the '+sip.rendering="no"' field parameter to indicate that it is putting the call on hold. ([ref-dialog-event] section 5.2)

3. The remote UA sends a 2xx to the re-INVITE, and includes an SDP offer giving its own listening address/port. If the remote UA understands the sip.rendering feature parameter, the offer may indicate that it will not send media by specifying the media directionalities as "recvonly" (the reverse of "on-hold") or

perhaps "inactive". But the remote UA may offer to send media.

4. The executing UA uses this offer to derive the offer SDP of an initial INVITE that it sends to the configured music-on-hold (MOH) source. The SDP in this request is largely copied from the SDP returned by the remote UA in the previous step, particularly regarding the provided listening address/port and payload type numbers. But the media directionalities are restricted to "recvonly" or "inactive" as appropriate. The executing UA may want or need to change the o= line. In addition, some a=rtpmap lines may need to be added to control the assignment of RTP payload type numbers.[Section 2.7]

5. The MOH source sends a 2xx response to the INVITE, which contains an SDP answer that should include its media source address as its listening address/port. This SDP must necessarily[offer-answer] specify "sendonly" or "inactive" as the directionality for all media streams. (Although this address/port should receive no RTP, by convention UAs use their declared RTP listening ports as

their RTP source ports as well. The answer SDP will reach the remote UA, thus informing it of the address/port from which the MOH media will come, and presumably preventing the remote UA from ignoring the MOH media as SPIT. This functionality requires the SDP answer to contain the sending address/port in the c= line, even though the MOH source does not receive RTP.)

6. The executing UA sends this SDP answer as its SDP answer in the ACK for the re-INVITE to the remote UA. The o= line in the answer must be modified to be within the sequence of o= lines previously generated by the executing UA in the dialog. Any dynamic payload type number assignments that have been created in the answer must be recorded in the state of the original dialog.

7. Due to the sip.rendering feature parameter in the Contact of the re-INVITE and the media directionality in the SDP answer contained in the ACK, the on-hold state of the dialog is established (at the executing end).

8. After this point, the MOH source generates RTP containing the music-on-hold media, and sends it directly to the listening address/port of the remote UA. The executing UA maintains two

dialogs (one to the remote UA, one to the MOH source), but does not see or handle the MOH RTP.

2.2. Taking a Call off Hold and Terminating the External Media Stream

1. The executing user instructs the executing UA to take the dialog off-hold.

2. The executing UA sends a re-INVITE to the remote UA with SDP that requests to receive media. The Contact header of the re-INVITE does not include the '+sip.rendering="no"' field parameter. (It may contain a sip.rendering field parameter with value "yes" or "unknown", or it may omit the field parameter.) Thus this INVITE removes the on-hold state of the dialog (at the executing end). (Note that the version in o= line of the offered SDP must account for the SDP versions that were passed through from the MOH source, and that any payload type numbers that were assigned in SDP provided by the MOH source must be respected.)

3. When the remote UA sends a 2xx response to the re-INVITE, the executing UA sends a BYE request in the dialog to the MOH source.

4. After this point, the MOH source does not generate RTP and ordinary RTP flow is re-established in the original dialog.

2.3. Example Message Flow

This section shows a message flow which is an example of this technique. The scenario is: Alice establishes a call with Bob. Bob then places the call on hold, with music-on-hold provided from an external source. Bob then takes the call off hold.

Note that this is just one possible message flow that illustrates this technique; numerous variations on these operations are allowed by the applicable standards.

```
Alice                 Bob       Music Source
```

Alice establishes the call:

```
   |                |              |
   |    INVITE F1   |              |
   |--------------->|              |
   | 180 Ringing F2 |              |
   |<---------------|              |
   |    200 OK F3   |              |
   |<---------------|              |
   |     ACK F4     |              |
   |--------------->|              |
   |       RTP      |              |
   |<==============>|              |
   |                |              |
```

Bob places Alice on hold, compelling Alice's UA to provide SDP:

```
   |                |              |
   |   INVITE F5    |              |
   |   (no SDP)     |              |
   |<---------------|              |
   |   200 OK F6    |              |
   |   (SDP offer)  |              |
   |--------------->|              |
   |                |              |
```

Bob's UA initiates music-on-hold:

```
   |                |              |
   |                |  INVITE F7   |
   |                |  (SDP offer, |
   |                |   rev. hold) |
   |                |------------->|
   |                | 200 OK F8    |
   |                | (SDP answer, |
```



```
   |                |  hold)       |
   |                |<-------------|
   |                |    ACK F9    |
   |                |------------->|
   |                |              |
```

Bob's UA provides an SDP answer containing the address/port
of the Music Source:

```
   |                 |              |
   | ACK F10         |              |
   | (SDP answer,    |              |
   |  hold           |              |
   |<----------------|              |
   |    no RTP       |              |
   |                 |              |
   |     Music-on-hold RTP          |
   |<===============================|
   |                 |              |
```

The music on hold is active.

Bob takes Alice off hold:

```
   |                 |              |
   |  INVITE F11     |              |
   |  (SDP offer)    |              |
   |<----------------|              |
   |   200 OK F12    |              |
   |   (SDP answer)  |              |
   |---------------->|              |
   |     ACK F13     |              |
   |<----------------|              |
   |                 |    BYE F14   |
   |                 |------------->|
   |                 |    200 F15   |
   |                 |<-------------|
   |       RTP       |              |
   |<===============>|              |
   |                 |              |
```

The normal media session between Alice and Bob is resumed.

```
/* Alice calls Bob. */

F1 INVITE Alice -> Bob
```



```
INVITE sips:bob@biloxi.example.com SIP/2.0
```

```
Via: SIP/2.0/TLS atlanta.example.com:5061
 ;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Alice <sips:alice@atlanta.example.com>;tag=1234567
To: Bob <sips:bob@biloxi.example.com>
Call-ID: 12345600@atlanta.example.com
CSeq: 1 INVITE
Contact: <sips:a8342043f@atlanta.example.com;gr>
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY
Supported: replaces, gruu
Content-Type: application/sdp
Content-Length: [omitted]

v=0
o=alice 2890844526 2890844526 IN IP4 atlanta.example.com
s=
c=IN IP4 atlanta.example.com
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000


F2 180 Ringing Bob -> Alice

SIP/2.0 180 Ringing
Via: SIP/2.0/TLS atlanta.example.com:5061
 ;branch=z9hG4bK74bf9
 ;received=192.0.2.103
From: Alice <sips:alice@atlanta.example.com>;tag=1234567
To: Bob <sips:bob@biloxi.example.com>;tag=23431
Call-ID: 12345600@atlanta.example.com
CSeq: 1 INVITE
Contact: <sips:bob@biloxi.example.com>
Content-Length: 0


F3 200 OK Bob -> Alice

SIP/2.0 200 OK
Via: SIP/2.0/TLS atlanta.example.com:5061
 ;branch=z9hG4bK74bf9
 ;received=192.0.2.103
From: Alice <sips:alice@atlanta.example.com>;tag=1234567
To: Bob <sips:bob@biloxi.example.com>;tag=23431
Call-ID: 12345600@atlanta.example.com
CSeq: 1 INVITE
Contact: <sips:bob@biloxi.example.com>
```

```
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY
Supported: replaces
Content-Type: application/sdp
Content-Length: [omitted]

v=0
o=bob 2890844527 2890844527 IN IP4 biloxi.example.com
s=
c=IN IP4 biloxi.example.com
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000


F4 ACK Alice -> Bob

ACK sips:bob@biloxi.example.com SIP/2.0
Via: SIP/2.0/TLS atlanta.example.com:5061
 ;branch=z9hG4bK74bfd
Max-Forwards: 70
From: Alice <sips:alice@atlanta.example.com>;tag=1234567
To: Bob <sips:bob@biloxi.example.com>;tag=23431
Call-ID: 12345600@atlanta.example.com
CSeq: 1 ACK
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY
Supported: replaces
Content-Length: 0


/* Bob places Alice on hold. */

/* The re-INVITE contains no SDP, thus compelling Alice's UA
   to provide an offer. */

F5 INVITE Bob -> Alice

INVITE sips:a8342043f@atlanta.example.com;gr SIP/2.0
Via: SIP/2.0/TLS biloxi.example.com:5061
 ;branch=z9hG4bK874bk
To: Alice <sips:alice@atlanta.example.com>;tag=1234567
From: Bob <sips:bob@biloxi.example.com>;tag=23431
Call-ID: 12345600@atlanta.example.com
CSeq: 712 INVITE
Contact: <sips:bob@biloxi.example.com>;+sip.rendering="no"
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY
Supported: replaces
```

```
Content-Length: 0
```



```
/* Alice's UA provides an SDP offer.
   Since it does not know that it is being put on hold,
   the offer is the same as the original offer and describes
   bidirectional media. */

F6 200 OK Alice -> Bob

SIP/2.0 200 OK
Via: SIP/2.0/TLS biloxi.example.com:5061
 ;branch=z9hG4bK874bk
 ;received=192.0.2.105
To: Alice <sips:alice@atlanta.example.com>;tag=1234567
From: Bob <sips:bob@biloxi.example.com>;tag=23431
Call-ID: 12345600@atlanta.example.com
CSeq: 712 INVITE
Contact: <sips:a8342043f@atlanta.example.com;gr>
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY
Supported: replaces, gruu
Content-Type: application/sdp
Content-Length: [omitted]

v=0
o=alice 2890844526 2890844526 IN IP4 atlanta.example.com
s=
c=IN IP4 atlanta.example.com
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000
a=active


/* Bob's UA initiates music-on-hold. */

/* This INVITE contains Alice's offer, but with the media
   direction set to "reverse hold", receive-only. */

F7 INVITE Bob -> Music Source

INVITE sips:music@source.example.com SIP/2.0
```

```
Via: SIP/2.0/TLS biloxi.example.com:5061
 ;branch=z9hG4bKnashds9
Max-Forwards: 70
From: Bob <sips:bob@biloxi.example.com>;tag=02134
To: Music Source <sips:music@source.example.com>
Call-ID: 4802029847@biloxi.example.com
CSeq: 1 INVITE
Contact: <sips:bob@biloxi.example.com>
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY
```



```
Supported: replaces, gruu
Content-Type: application/sdp
Content-Length: [omitted]

v=0
o=bob 2890844534 2890844534 IN IP4 atlanta.example.com
s=
c=IN IP4 atlanta.example.com
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000
a=recvonly
```

F8 200 OK Music Source -> Bob

```
SIP/2.0 200 OK
Via: SIP/2.0/TLS biloxi.example.com:5061
 ;branch=z9hG4bKnashds9
 ;received=192.0.2.105
From: Bob <sips:bob@biloxi.example.com>;tag=02134
To: Music Source <sips:music@source.example.com>;tag=56323
Call-ID: 4802029847@biloxi.example.com
Contact: <sips:music@source.example.com>
CSeq: 1 INVITE
Content-Length: [omitted]

v=0
o=MusicSource 2890844576 2890844576 IN IP4 source.example.com
s=
c=IN IP4 source.example.com
t=0 0
```

```
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000
a=sendonly


F9 ACK Bob -> Music Source

ACK sips:music@source.example.com SIP/2.0
Via: SIP/2.0/TLS source.example.com:5061
 ;branch=z9hG4bK74bT6
From: Bob <sips:bob@biloxi.example.com>;tag=02134
To: Music Source <sips:music@source.example.com>;tag=56323
Max-Forwards: 70
Call-ID: 4802029847@biloxi.example.com
CSeq: 1 ACK
Content-Length: 0
```



```
/* Bob's UA now sends the ACK that completes the re-INVITE
   to Alice and completes the SDP offer/answer.
   The ACK contains the SDP received from the Music Source,
   and thus contains the address/port from which the Music Source
   will send media. */

F10 ACK Bob -> Alice

ACK sips:a8342043f@atlanta.example.com;gr SIP/2.0
Via: SIP/2.0/TLS biloxi.example.com:5061
 ;branch=z9hG4bKq874b
To: Alice <sips:alice@atlanta.example.com>;tag=1234567
From: Bob <sips:bob@biloxi.example.com>;tag=23431
Call-ID: 12345600@atlanta.example.com
CSeq: 712 ACK
Contact: <sips:bob@biloxi.example.com>;+sip.rendering="no"
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY
Supported: replaces
Content-Length: [omitted]

v=0
o=bob 2890844527 2890844528 IN IP4 biloxi.example.com
s=
c=IN IP4 source.example.com
t=0 0
```

```
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000
a=sendonly

/* Bob picks up the call by sending a re-INVITE to Alice. */

F11 INVITE Bob -> Alice

INVITE sips:a8342043f@atlanta.example.com;gr SIP/2.0
Via: SIP/2.0/TLS biloxi.example.com:5061
 ;branch=z9hG4bK874bk
To: Alice <sips:alice@atlanta.example.com>;tag=1234567
From: Bob <sips:bob@biloxi.example.com>;tag=23431
Call-ID: 12345600@atlanta.example.com
CSeq: 713 INVITE
Contact: <sips:bob@biloxi.example.com>
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY
Supported: replaces
Content-Type: application/sdp
Content-Length: [omitted]

v=0
o=bob 2890844527 2890844529 IN IP4 biloxi.example.com
```



```
s=
c=IN IP4 biloxi.example.com
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000


F12 200 OK Alice -> Bob

SIP/2.0 200 OK
Via: SIP/2.0/TLS biloxi.example.com:5061
 ;branch=z9hG4bK874bk
 ;received=192.0.2.105
To: Alice <sips:alice@atlanta.example.com>;tag=1234567
From: Bob <sips:bob@biloxi.example.com>;tag=23431
Call-ID: 12345600@atlanta.example.com
CSeq: 713 INVITE
Contact: <sips:a8342043f@atlanta.example.com;gr>
```

```
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY
Supported: replaces, gruu
Content-Type: application/sdp
Content-Length: [omitted]

v=0
o=alice 2890844526 2890844527 IN IP4 atlanta.example.com
s=
c=IN IP4 atlanta.example.com
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

F13 ACK Bob -> Alice

```
ACK sips:a8342043f@atlanta.example.com;gr SIP/2.0
Via: SIP/2.0/TLS biloxi.example.com:5061
 ;branch=z9hG4bKq874b
To: Alice <sips:alice@atlanta.example.com>;tag=1234567
From: Bob <sips:bob@biloxi.example.com>;tag=23431
Call-ID: 12345600@atlanta.example.com
CSeq: 713 ACK
Contact: <sips:bob@biloxi.example.com>
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY
Supported: replaces
Content-Length: 0
```

F14 BYE Bob -> Music Source


```
BYE sips:music@source.example.com SIP/2.0
Via: SIP/2.0/TLS biloxi.example.com:5061
 ;branch=z9hG4bK74rf
Max-Forwards: 70
From: Bob <sips:bob@biloxi.example.com>;tag=02134
To: Music Source <sips:music@source.example.com>;tag=56323
Call-ID: 4802029847@biloxi.example.com
CSeq: 2 BYE
Contact: <sips:bob@biloxi.example.com>
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY
Supported: replaces, gruu
```

```
Content-Length: [omitted]


F15 200 OK Music Source -> Alice

SIP/2.0 200 OK
Via: SIP/2.0/TLS atlanta.example.com:5061
 ;branch=z9hG4bK74rf
 ;received=192.0.2.103
From: Bob <sips:bob@biloxi.example.com>;tag=02134
To: Music Source <sips:music@source.example.com>;tag=56323
Call-ID: 4802029847@biloxi.example.com
CSeq: 2 BYE
Content-Length: 0


/* Normal media session between Alice and Bob is resumed */
```

2.4. Re-INVITE and UPDATE from the Remote UA

While the call is on-hold, the remote UA can send a request to modify the SDP or the feature parameters of its Contact header. This can be done with either an INVITE or UPDATE method, both of which have much the same effect in regard to MOH.

A common reason for a re-INVITE will be when the remote UA desires to put the dialog on hold on its end. And because of the need to support this case, an implementation must process INVITEs and UPDATEs during the on-hold state as described below.

The executing UA handles these requests by echoing requests and responses: an incoming request from the remote UA causes the executing UA to send a similar request to the MOH source and an incoming response from the MOH source causes the executing UA to send a similar response to the remote UA. In all cases, SDP offers or answers that are received are added as bodies to the stimulated request or response to the other UA.

The passed-through SDP will usually need its o= line modified. The directionality attributes may need to be restricted. In regard to payload type numbers, since the mapping has already been established within the MOH dialog, a=rtpmap lines need not be added.

2.5. INVITE with Replaces

The executing UA must be prepared to receive INVITE requests with a Replaces headers that replaces the original dialog, and similarly it must be prepared to receive REFER requests within the dialog. The SDP within the new dialog is negotiated by being passed through to the MOH source within a new dialog with the MOH source. The SDP offer or answer can be passed to the MOH source with only modification to the o= line and directionality attributes.

In some cases, the previous dialog with the MOH source can be reused, but only if the executing UA presents the first offer within the new dialog, as otherwise there is no way to force the RTP payload types that have been used previously in the MOH dialog to be mapped to the correct codecs in the new dialog.

2.6. Re-INVITE and UPDATE from the Music-On-Hold Source

It is possible for the MOH source to send an INVITE or UPDATE request, and the executing UA can support doing so in similar manner as requests from the remote UA. However, if the MOH source is within the same administrative domain as the executing UA, the executing UA may have knowledge that the MOH source will not (or need not) make such requests, and so can respond to any such request with a failure response, avoiding the need to pass the request through.

However, in an environment in which ICE[ice] is supported, the MOH source may need to send requests as part of ICE negotiation[elwell] with the remote UA. Hence, in environments that support ICE, the executing UA must be able to pass through requests from the MOH source as well as requests from the remote UA.

Again, as SDP is passed through, its o= line will need to be modified. In some cases, the directionality attributes will need to be restricted.

2.7. Payload Type Numbers

In this technique, the MOH source generates an SDP answer that the executing UA presents to the remote UA as an answer within the original dialog. In basic functionality, this presents no problem, because [offer-answer] (section 6.1, at the very end) specifies that the payload type numbers used in either direction of RTP are the ones specified in the SDP sent by the recipient of the RTP.

But strict compliance to [offer-answer] (section 8.3.2) requires that payload type numbers used in SDP may only duplicate the payload type numbers used in any SDP used in the same direction in the dialog if the payload type numbers represent the same media format (codec) as they did previously. However, the MOH source has no knowledge of the payload type numbers previously used in the original dialog, and it may accidentally specify a media format for a previously used payload type number in its answer (or in a subsequently generated INVITE or UPDATE). This would cause no problem with media decoding, as it cannot send any format that was not in the remote UA's offer, but it would violate [offer-answer].

Strictly speaking, it is impossible to avoid this problem because the generator of a first answer in its dialog can choose the payload numbers independently of the payload numbers in the offer, and the MOH server believes that its answer is first in the dialog. Thus the only absolute solution is to have the executing UA rewrite the SDP that passes through it to reassign payload type numbers, which would also require it to rewrite the payload type numbers in the RTP packets -- a very undesirable solution. But we can exploit a SHOULD-level requirement in [offer-answer] (section 6.1): "In the case of RTP, if a particular codec was referenced with a specific payload type number in the offer, that same payload type number SHOULD be used for that codec in the answer." If the MOH source obeys this restriction, the executing UA can modify the offer SDP to "reserve" all payload type numbers that have ever been offered by the executing UA to prevent the MOH source from using them for different media formats.

When the executing UA is composing the INVITE to the MOH source, it compiles a list of all the (dynamically-assigned) payload type numbers which have been used by it (or by MOH sources on its behalf) in the original dialog but which are not mapped to a media format in the current offer SDP. (The executing UA must be maintaining a list of all previously used payload type numbers anyway, in order to comply with [offer-answer].) Then, for each of these payload type numbers, it inserts session-level or media-level (as appropriate) a=rtpmap lines specifying the payload type number and the media format that it has been used for. Because of the reuse rule, the MOH source SHOULD not propose those payload type numbers for any other media format.

Note that any re-INVITEs from the remote UA that the executing UA passes through to the MOH server require similar modification, as payload type numbers that the MOH server receives in past offers are not absolutely reserved against its use (as they have not been sent

in SDP by the MOH server) nor is there a SHOULD-level proscription against using them in the current answer (as they do not appear in the current offer).

This should provide an adequate solution to the problems with payload type numbers, as it will fail only if (1) the remote UA is particular that other UAs follow the rule about not re-defining payload type numbers, and (2) the MOH server does not follow the SHOULD-level requirement of [offer-answer] section 6.1.

Let us show how this process works by modifying the example Section 2.3 with this specific assignment of supported codecs:

Alice supports formats X and Y

Bob supports formats X and Z

Music Source supports formats Y and Z

In this case, the SDP exchanges are:

F1 offers X and Y, F3 answers X and Z (which cannot be used)

F6 offers X and Y, but F7 offers X, Y, and a place-holder to block type 92

F8/F10 answers Y

```
F1 INVITE Alice -> Bob

INVITE sips:bob@biloxi.example.com SIP/2.0
Via: SIP/2.0/TLS atlanta.example.com:5061
 ;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Alice <sips:alice@atlanta.example.com>;tag=1234567
To: Bob <sips:bob@biloxi.example.com>
Call-ID: 12345600@atlanta.example.com
CSeq: 1 INVITE
Contact: <sips:a8342043f@atlanta.example.com;gr>
```

```
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY
Supported: replaces, gruu
Content-Type: application/sdp
Content-Length: [omitted]

v=0
o=alice 2890844526 2890844526 IN IP4 atlanta.example.com
s=
```



```
c=IN IP4 atlanta.example.com
t=0 0
m=audio 49170 RTP/AVP 90 91
a=rtpmap:90 X/8000
a=rtpmap:91 Y/8000
```

F3 200 OK Bob -> Alice

```
SIP/2.0 200 OK
Via: SIP/2.0/TLS atlanta.example.com:5061
 ;branch=z9hG4bK74bf9
 ;received=192.0.2.103
From: Alice <sips:alice@atlanta.example.com>;tag=1234567
To: Bob <sips:bob@biloxi.example.com>;tag=23431
Call-ID: 12345600@atlanta.example.com
CSeq: 1 INVITE
Contact: <sips:bob@biloxi.example.com>
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY
Supported: replaces
Content-Type: application/sdp
Content-Length: [omitted]

v=0
o=bob 2890844527 2890844527 IN IP4 biloxi.example.com
s=
c=IN IP4 biloxi.example.com
t=0 0
m=audio 3456 RTP/AVP 90 92
a=rtpmap:90 X/8000
a=rtpmap:92 Z/8000
```

```
F6 200 OK Alice -> Bob

SIP/2.0 200 OK
Via: SIP/2.0/TLS biloxi.example.com:5061
 ;branch=z9hG4bK874bk
 ;received=192.0.2.105
To: Alice <sips:alice@atlanta.example.com>;tag=1234567
From: Bob <sips:bob@biloxi.example.com>;tag=23431
Call-ID: 12345600@atlanta.example.com
CSeq: 712 INVITE
Contact: <sips:a8342043f@atlanta.example.com;gr>
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY
Supported: replaces, gruu
Content-Type: application/sdp
Content-Length: [omitted]
```



```
v=0
o=alice 2890844526 2890844526 IN IP4 atlanta.example.com
s=
c=IN IP4 atlanta.example.com
t=0 0
m=audio 49170 RTP/AVP 90 91
a=rtpmap:90 X/8000
a=rtpmap:91 Y/8000


F7 INVITE Bob -> Music Source

INVITE sips:music@source.example.com SIP/2.0
Via: SIP/2.0/TLS biloxi.example.com:5061
 ;branch=z9hG4bKnashds9
Max-Forwards: 70
From: Bob <sips:bob@biloxi.example.com>;tag=02134
To: Music Source <sips:music@source.example.com>
Call-ID: 4802029847@biloxi.example.com
CSeq: 1 INVITE
Contact: <sips:bob@biloxi.example.com>
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY
Supported: replaces, gruu
Content-Type: application/sdp
Content-Length: [omitted]
```

```
v=0
o=bob 2890844534 2890844534 IN IP4 atlanta.example.com
s=
c=IN IP4 atlanta.example.com
t=0 0
m=audio 49170 RTP/AVP 0
m=audio 49170 RTP/AVP 90 91 92
a=rtpmap:90 X/8000
a=rtpmap:91 Y/8000
a=rtpmap:92 x-reserved/8000
a=recvonly
```

F8 200 OK Music Source -> Bob

```
SIP/2.0 200 OK
Via: SIP/2.0/TLS biloxi.example.com:5061
 ;branch=z9hG4bKnashds9
 ;received=192.0.2.105
From: Bob <sips:bob@biloxi.example.com>;tag=02134
To: Music Source <sips:music@source.example.com>;tag=56323
Call-ID: 4802029847@biloxi.example.com
```



```
Contact: <sips:music@source.example.com>
CSeq: 1 INVITE
Content-Length: [omitted]

v=0
o=MusicSource 2890844576 2890844576 IN IP4 source.example.com
s=
c=IN IP4 source.example.com
t=0 0
m=audio 49170 RTP/AVP 91
a=rtpmap:91 Y/8000
a=sendonly
```

3. Advantages

This technique for providing music-on-hold has advantages over other methods now in use:

1. The original dialog is not transferred to another UA, so the "remote endpoint URI" displayed by the remote endpoint's user interface and dialog event package[ref-dialog-event] does not change during the call.[service-examples]

2. The music-on-hold media are sent directly from the music-on-hold source to the remote UA, rather than being relayed through the

executing UA.

3. The remote UA sees, in the incoming SDP, the address/port that the MOH source will send MOH media from, thus allowing it to render the media, even if it is filtering incoming media based on originating address as a SPIT preventative.

4. The technique requires relatively simple manipulation of SDP, and in particular: (1) does not require a SIP element to modify unrelated SDP to be acceptable to be sent within an already established sequence of SDP (a problem with [service-examples-11]), and (2) does not require converting an SDP answer into an SDP offer (which was a problem with the -00 version of this document, as well as with [service-examples-11]).

5. It complies with the payload type number rules.[offer-answer]

4. The Importance of Offering All Available Media Formats

Failures can happen if SDP offerers do not always offer all media formats that they support. Doing so is considered best practice, but some elements will offer only formats that have already been in use in the dialog.

An example of how omitting media formats in an offer can lead to failure is as follows: Suppose that the UAs in Section 2.3 each support the following media formats:

Alice supports formats X and Y

Bob supports formats X and Z

Music Source supports formats Y and Z

In this case, the SDP exchanges are:

F1 offers X and Y, F3 answers X

F6/F7 offers X and Y, F8/F10 answers Y

F11 offers X and Z, F12 answers X

Note that in exchange 2, if Alice assumes that because only format X is in use that she should offer only X, the exchange fails. In exchange 3, Bob offers formats X and Z, even though neither is in use at the time (because Bob is not involved in the media streams).

5. Security Considerations

Some UAs filter incoming media based on the address of origin in order to avoid SPIT. The technique described in this document ensures that any UA that should render MOH media will be informed of the source address of the media via the SDP that it receives. This should allow such UAs to filter without interfering with MOH operation.

6. Acknowledgments

The original version of this proposal was derived from [service-examples-11] and the similar implementation of MOH in the Snom UA. Significant improvements to the sequence of operations, allowing improvements to the SDP handling, were suggested by Venkatesh[venkatesh].

John Elwell[elwell] pointed out the need for the executing UA to pass through re-INVITEs/UPDATEs in order to allow ICE negotiation.

Paul Kyzivat[kyzivat] pointed out the difficulties regarding re-use of payload type numbers.

Paul Kyzivat suggested adding sectionSection 4 showing why offerers should always include all supported formats.

7. Revision History

7.1. Changes from draft-worley-service-example-00 to draft-worley-service-example-01

Removed the original "Example Message Flow" and promoted the "Alternative Example Message Flow" to replace it because of a number of flaws that were found during the discussion of -00 on the SIPPING mailing list.

Described the use of the sip.rendering feature parameter to indicate on-hold status.

7.2. Changes from draft-worley-service-example-01 to draft-worley-service-example-02

Added discussion of passing though re-INVITEs and UPDATEs.

Added discussion of payload type numbers.

Added Acknowledgments section.

7.3. Changes from draft-worley-service-example-02 to draft-worley-service-example-03

Added sectionSection 4 showing the importance of the offerer always including all supported media formats.

Updated references.

Revised handling of payload type numbers when passing offer to MOH serverSection 2.7 based on observations by Paul Kyzivat.

8. References

8.1. Normative References

[sip] 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.

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

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

8.2. Informative References

[service-examples] Johnston, A., Sparks, R., Cunningham, C., Donovan, S., and K. Summers, "Session Initiation Protocol Service Examples", RFC 5359, October 200.

[service-examples-11] Johnston, A., Sparks, R., Cunningham, C., Donovan, S., and K. Summers, "Session Initiation Protocol Service Examples", I-D draft-ietf-sipping-service-examples-11, October 2006.

[ref-dialog-event]
Rosenberg, J., Schulzrinne, H., and R. Mahy, "An INVITE-Initiated Dialog Event Package for the Session Initiation Protocol (SIP)", RFC 4235, November 2005.

[venkatesh]
Venkatesh, "Subject: Re: [Sipping] I-D ACTION:draft-ietf-sipping-service-examples-11.txt", IETF Sipping mailing list msg12180, October 2006.

[ice] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", I-D draft-ietf-mmusic-ice-19, October 2007.

[elwell] Elwell, J., "Subject: [Sipping] RE: I-D Action:draft-worley-service-example-00.txt", IETF Sipping mailing list msg14678, November 2007.

[kyzivat] Kyzivat, P., "Subject: Re: [Sipping] I-D ACTION:draft-ietf-sipping-service-examples-11.txt", IETF Sipping mailing list msg12181, October 2006.

[offer-answer-bis]
Sawada, T. and P. Kyzivat, "SIP (Session Initiation Protocol) Usage of the Offer/Answer Model", I-D draft-ietf-sipping-sip-offeranswer-10, January 2009.

Author's Address

Dale R. Worley
Nortel Networks Corp.
600 Technology Park Dr.
Billerica, MA 01821
US

Email: dworley@nortel.com
URI: http://www.nortel.com