Internet Engineering Task Force — SIP WG
INTERNET-DRAFT
draft-camarillo-sip-deaf-01.ps
G. Camarillo, E. Burger, H. Schulzrinne, A. van Wijk
Ericsson, SnowShore Networks, Columbia University, Ericsson
November 4, 2002
Expires: May, 2003

Transcoding Services Invocation in the Session Initiation Protocol

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

To view the list Internet-Draft Shadow Directories, see http://www.ietf.org/shadow.html.

Copyright Notice

Copyright (c) The Internet Society (2002). All Rights Reserved.

Abstract

This document describes how to use SIP third party call control to invoke transcoding services that involve media manipulations by a media server. In particular, this document describes how to meet the requirements for the Session Initiation Protocol in support of deaf, hard of hearing and speech-impaired individuals using third party call control.

Contents

1 Introduction

A key requirement for SIP [1] to support deaf, hard of hearing and speech-impaired individuals [2] is to be able to introduce intermediaries that provide transcoding services to a session. Common examples of transcoding services are speech-to-text and speech-to-SL (Sign Language), a video format. One can envision other transcoding scenarios, such as a hearing-impaired user that wishes to still see and hear the original media (voice and video), yet also see a text translation and sign language feed.

Transcoding can be symmetric, as in a speech-to-text plus text-to-speech. This would be the case for a TTY user communicating with a speaking and hearing user. Transcoding can also be asymmetric, as in a one-way speech-to-text transcoding. This would be the case for a hearing impaired user that can still talk.

Note that the mechanisms described in this document are not specific to the support of deaf, hard of hearing and speech-impaired individuals. If they are used to invoke other types of transcoding services, such as video-to-audio or image-to-audio, they can be used to support blind individuals as well. In fact, this document describes a general mechanism for invoking any service that involves media manipulations by a media server.

One example of this is wireless communications. Many network topologies require transcoding between different media formats. For example, 3GPP handsets may use GSM AMR2, an advanced audio codec, while older handsets may use GSM FR, SVC, or other codecs. Existing wireless topologies transcode all media to a common format, thus introducing a number of transcoding steps that is often unnecessary. Thus there is a need to transcode between these codecs on an ad hoc, or on demand, basis.

This document does not describe media server discovery. That is an orthogonal problem that one can address using user agent provisioning, registrars, or other methods.

There are two models for invoking a transcoding service. The first is to use an RTP mixer (conference bridge) that negotiates the appropriate media parameters on each individual leg. The second is to use third party call control [3], also referred to as 3pcc, to invoke the transcoding service. As the conference bridge model is straightforward and standard as is, this document will only describe how it works. Section 2 describes the conference bridge model and Section 3 describes the 3pcc model.

All the examples provided in this document use the Session Description Protocol (SDP) [4]. However, other session description formats can be used with the same call flows.

2 Conference Bridge Transcoding Model

Invoking transcoding services from a server (T) for a session between two user agents (A and B) involves establishing two media sessions; one between A and T and another between T and B. How to invoke T's services (i.e., how to establish both A-T and T-B sessions) depends on how we model the transcoding service. We have considered two possible models, namely, the conferencing server model and a model specific to transcoding services.

A conference server typically establishes an audio stream with each participant of a conference. The server sends over each individual stream the media received over the rest of the streams, typically performing some mixing. The conference server may have to send audio to different participants using different audio codecs. Each of these audio streams is typically established though an INVITE request from each participant to the conference server that carries a session description.

We can think of a transcoding service as a two-party conference server that not only changes the codec in use, but the format of the media as well (e.g., audio to text). Using this model, the whole A-T-B session would be established in the same way as a conference. T would receive two INVITEs; one INVITE with A's session description and another with B's session description, as shown in Figure 1.

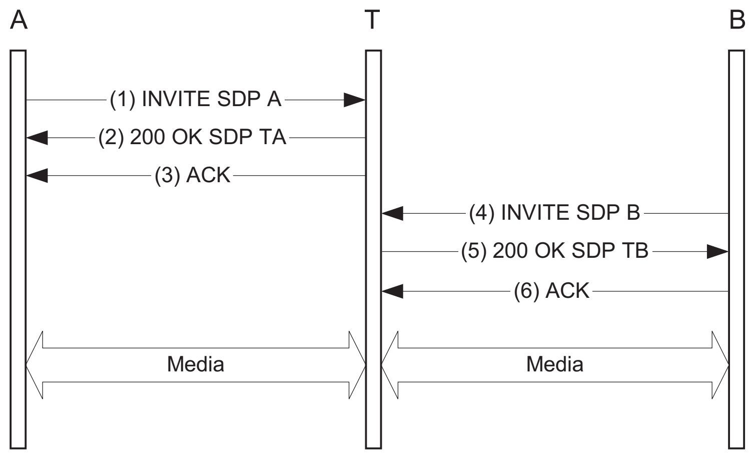

Figure 1: Conference bridge transcoding model

2.1 Back-To-Back User Agent Transcoding Model

The Back-To-Back User Agent (B2BUA) model is a particular case of the conferencing bridge model described above. Like other models for establishing a transcoding session, this model establishes two sessions, T-A and T-B. However, the establishment of T-B is initiated by T, as shown in Figure 2. The B2BUA can be thought of as a dial-in/dial-out conference bridge.

In this scenario, the user contacts a transcoding service URI using a Route header field in the INVITE request (1). That URI resolves to the B2BUA. The B2BUA determines the need and method for transcoding the media streams. How T knows what media types B will find acceptable is beyond the scope of this document. A provides T with the final destination of the session (i.e., B) in the Request-URI of the INVITE (1). B replies with its SDP (3). T then responds to A's original INVITE (1) with the transcoded representation of B's SDP (4). A ACKs the 200 OK (5) and T ACKs B's 200 OK (6). At this point we have the transcoding session established.

Note that using multiple Route entries in the initial INVITE, it is possible to traverse multiple transcoding services before reaching the final destination.

2.2 Application Server Transcoding Model

The transcoding service T in Figure 2 handles both media and SIP signalling. However, a real implementation of this service would typically separate the B2BUA from the transcoding engine, using a protocol between them. Various protocols can be used on this interface, but we recommend

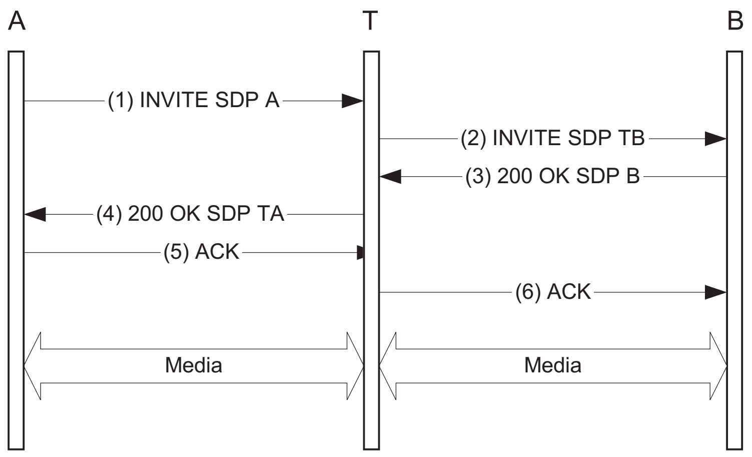

Figure 2: B2BUA transcoding model

to use SIP. In this case, the B2BUA would be an application server using 3pcc to control the transcoding service. The resulting call flow is very similar to the one shown in Figure 5. The B2BUA acting as the application server would play A's role invoking the transcoding service T. Therefore, the application server transcoding model is a mixture of the conference bridge transcoding model and the 3pcc transcoding model.

The conference bridge, B2BUA and application server transcoding models have the the advantage of being general, since the transcoding service is modeled as a particular case of a conference. Using these models provides a clean way to invoke transcoding services for conferences where many users and transcoding instances are involved. Although this is an interesting situation, our expectation is that the vast majority of transcoding for a given session will occur at a single media server between two users. In this case, 3pcc is a better alternative. 3pcc provides better fail-over mechanisms in case of failure in the conference bridge or in the B2BUA.

3 3PCC Transcoding Model

If we model T as a transcoding service rather than a special case of a conferencing server, a single INVITE transaction would provide T with both A's and B's session descriptions. In order to provide in a single session description information about media streams that belong to different entities (A and B), the session description format in use should provide a means to define how these streams should be mapped. For instance, in a session description with two audio streams and one text stream, a possible mapping would be the following; the information received over the first audio stream should be sent over the text stream and over the second audio stream, and the incoming text should be sent only over the first audio stream. SDP [4] can convey this information using the source and sink attributes [5].

As stated previously, the invocation of a transcoding service consists of establishing two sessions; A-T and T-B. How these sessions are established depends on which party, the caller (A) or the called party (B), invokes the transcoding services. However, we have followed a general principle to design our 3pcc flows. A 200 OK response from the transcoding service have to be received before contacting the called party. This tries to ensure that the transcoding service will be available when the called party accepts the session.

However, note that the transcoding service does not know the exact type of transcoding it will be performing until the called party accepts the session. Therefore, there are always changes of failing to provide transcoding services after the called party has accepted the session. A system with tough requirements could use preconditions to avoid this situation. When preconditions are used, the called party is not alerted until everything is ready for the session.

All the figures in this document follow the naming convention below:

SDP A: A session description generated by A. It contains, among other things, the transport address/es (IP address and port number) where A wants to receive media for each particular stream.

SDP B: A session description generated by B. It contains, among other things, the transport address/es where B wants to receive media for each particular stream.

SDP A+B: A session description that contains, among other things, the transport address/es where A wants to receive media and the transport address/es where B wants to receive media.

SDP TA: A session description generated by T and intended for A. It contains, among other things, the transport address/es where T wants to receive media from A.

SDP TB: A session description generated by T and intended for B. It contains, among other things, the transport address/es where T wants to receive media from B.

SDP TA+TB: A session description generated by T that contains, among other things, the transport address/es where T wants to receive media from A and the transport address/es where T wants to receive media from B.

3.1 Called Party Invocation

In this scenario B receives an INVITE from A, and B decides to introduce T in the session. Figure 3 shows the call flow for this scenario.

In Figure 3 A can both hear and speak and B is a deaf user with a speech impairment. A proposes to establish a session that consists of an audio stream (1). B wants to send and receive only text, so it invokes a transcoding service T that will perform both speech-to-text and text-to-speech conversions (2). The session descriptions of Figure 3 are partially shown below.

(1) INVITE SDP A

```
m=audio 20000 RTP/AVP 0
c=IN IP4 A.domain.com
```

(2) INVITE SDP A+B

```
m=audio 20000 RTP/AVP 0
c=IN IP4 A.domain.com
a=source:1
```

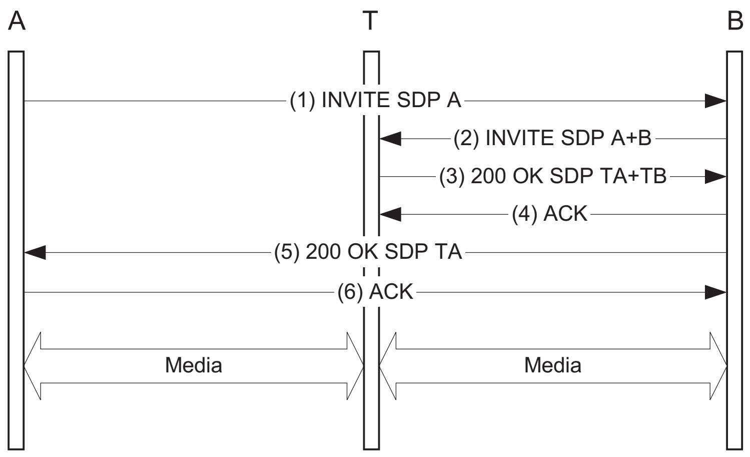

Figure 3: Callee invocation of a transcoding service

```
a=sink:2
m=text 40000 RTP/AVP 96
c=IN IP4 B.domain.com
a=rtpmap:96 t140/1000
a=source:2
a=sink:1
```

(3) 200 OK SDP TA+TB

```
m=audio 30000 RTP/AVP 0
c=IN IP4 T.domain.com
a=source:1
a=sink:2
m=text 30002 RTP/AVP 96
c=IN IP4 T.domain.com
a=rtpmap:96 t140/1000
a=source:2
a=sink:1
```

(5) 200 OK SDP TA

```
m=audio 30000 RTP/AVP 0
c=IN IP4 T.domain.com
```

Four media streams (i.e., two bi-directional streams) have been established at this point:

1. Audio from A to T.domain.com:30000

2. Text from T to B.domain.com:40000

3. Text from B to T.domain.com:30002

4. Audio from T to A.domain.com:20000

When either A or B decide to terminate the session, B will send a BYE to T indicating that the session is over.

If the first INVITE (1) received by B is empty (no session description), the call flow is slightly different. Figure 4 shows the messages involved.

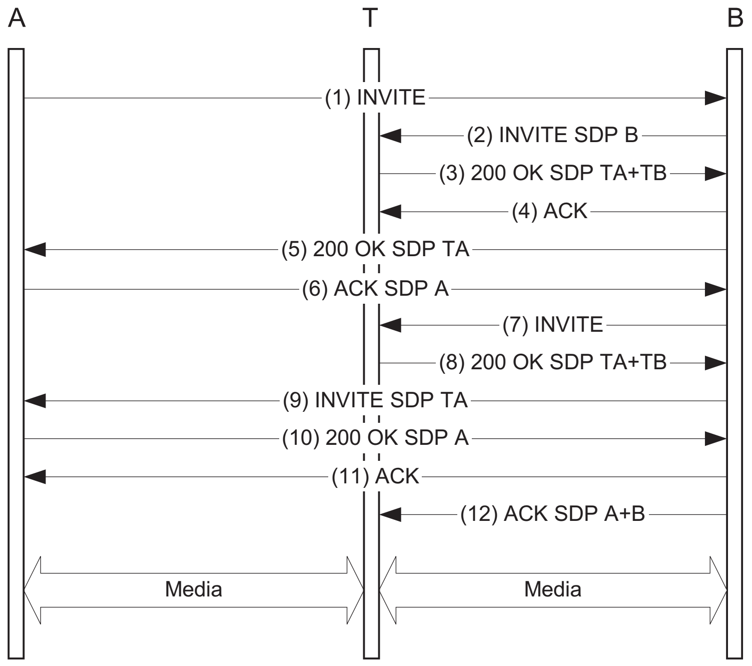

Figure 4: Callee invocation after initial INVITE without SDP

B may have different reasons for invoking T before knowing A's session description. B may want to hide its capabilities, and therefore it wants to return a session description with all the codecs B supports plus all the codecs T supports. Or T may provide recording services (besides transcoding), and B wants T to record the conversation, regardless of whether or not transcoding is needed.

This scenario is a bit more complex than the previous one. In INVITE (2), B still does not have SDP A, so it cannot provide T with that information. When B finally receives SDP A in (6), it has to send it to T. B sends an empty INVITE to T (7) and gets a 200 OK with SDP TA+TB (8). In general, this SDP TA+TB can be different than the one that was sent in (3). That is why B needs to send the updated SDP TA to A in (9). A then sends a possibly updated SDP A (10) and B sends it to T in (12). However, if T happens to return the same SDP TA+TB in (8) as in (3), B can skip messages (9), (10) and (11). Therefore, implementors of transcoding services are encouraged to return the same session description in (8) as in (3) in this type of scenario. The session descriptions of this flow are shown below:

(2) INVITE SDP A+B

```
m=audio 20000 RTP/AVP 0
```

```
c=IN IP4 0.0.0.0
a=source:1
a=sink:2
m=text 40000 RTP/AVP 96
c=IN IP4 B.domain.com
a=rtpmap:96 t140/1000
a=source:2
a=sink:1
```

(3) 200 OK SDP TA+TB

```
m=audio 30000 RTP/AVP 0
c=IN IP4 T.domain.com
a=source:1
a=sink:2
m=text 30002 RTP/AVP 96
c=IN IP4 T.domain.com
a=rtpmap:96 t140/1000
a=source:2
a=sink:1
```

(5) 200 OK SDP TA

```
m=audio 30000 RTP/AVP 0
c=IN IP4 T.domain.com
```

(6) ACK SDP A

```
m=audio 20000 RTP/AVP 0
c=IN IP4 A.domain.com
```

(8) 200 OK SDP TA+TB

```
m=audio 30004 RTP/AVP 0
c=IN IP4 T.domain.com
a=source:1
a=sink:2
m=text 30006 RTP/AVP 96
c=IN IP4 T.domain.com
a=rtpmap:96 t140/1000
a=source:2
a=sink:1
```

(9) INVITE SDP TA

```
m=audio 30004 RTP/AVP 0
```

```
c=IN IP4 T.domain.com
```

(10) 200 OK SDP A

```
m=audio 20002 RTP/AVP 0
c=IN IP4 A.domain.com
```

(12) ACK SDP A+B

```
m=audio 20002 RTP/AVP 0
c=IN IP4 A.domain.com
a=source:1
a=sink:2
m=text 40000 RTP/AVP 96
c=IN IP4 B.domain.com
a=rtpmap:96 t140/1000
a=source:2
a=sink:1
```

Four media streams (i.e., two bi-directional streams) have been established at this point:

1. Audio from A to T.domain.com:30004
2. Text from T to B.domain.com:40000
3. Text from B to T.domain.com:30006
4. Audio from T to A.domain.com:20002

3.2 Caller invocation

In this scenario A wishes to establish a session with B using a transcoding service. A uses 3pcc to set up the session between T and B. A may have different reasons for invoking T's services before knowing B's session description. A may have contacted B right before (i.e., INVITE-488 Not Acceptable Here-ACK)and noticed that they do not have any codecs in common. T may also provide recording services, which B wants to invoke.

The call flow we provide here is slightly different than the ones in [3]. In [3], the controller establishes a session between two user agents, being the user agents the ones deciding the characteristics of the streams. Here, A wants to establish a session between T and B, but A wants to decide how many and which types of streams are established. That is why A sends its session description in the first INVITE (1) to T, as opposed to the media-less initial INVITE recommended by [3]. Figure 5 shows the call flow for this scenario.

We do not include the session descriptions of this flow, since they are very similar to the ones in Figure 4. In this flow, if T returns the same SDP TA+TB in (8) as in (2), messages (9), (10) and (11) can be skipped.

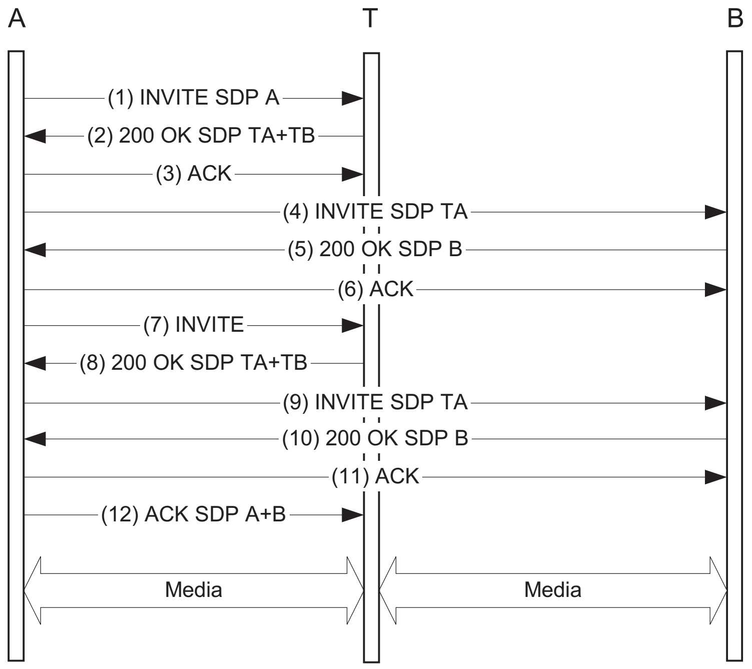

Figure 5: Caller invocation of a transcoding service

3.3 Receiving the original stream

Sometimes, as pointed out in the requirements for SIP in support of deaf, hard of hearing and speech-impaired individuals [2], a user wants to receive both the original stream (e.g., audio) and the transcoded stream (e.g., the output of the speech-to-text conversion). There are various possible solutions for this problem. One solution consists of using the SDP group attribute with FID semantics [6]. FID allows requesting that a stream is sent to two different transport addresses in parallel, as shown below:

```
a=group:FID 1 2
m=audio 20000 RTP/AVP 0
c=IN IP4 A.domain.com
a=mid:1
m=audio 30000 RTP/AVP 0
c=IN IP4 T.domain.com
a=mid:2
```

The problem with this solution is that the majority of the SIP user agents do not support FID. And even if FID is supported, many user agents do not support sending simultaneous copies of the same media stream at the same time.

Therefore, for user agents that do not support FID, requesting T to replicate the stream will always work. The following session description requests T to perform speech-to-text and text-to-speech conversions between the first audio stream and the text stream. In addition, it requests T to copy of the first audio stream to the second audio stream and send it to A.

```
m=audio 40000 RTP/AVP 0
c=IN IP4 B.domain.com
a=source:1
a=sink:2
m=audio 20000 RTP/AVP 0
c=IN IP4 A.domain.com
a=recvonly
a=sink:1
m=text 20002 RTP/AVP 96
c=IN IP4 A.domain.com
a=rtpmap:96 t140/1000
a=source:2
a=sink:1
```

3.4 Transcoding services in parallel

Transcoding services sometimes consist of human relays (e.g., a person performing speech-to-text and text-to-speech conversions for a session). If the same person is involved in both conversions (i.e., from A to B and from B to A), he or she has access to all the conversation. In order to provide some degree of privacy, sometimes two different persons are allocated to do the job (i.e., one person handles A-¿B and the other B-¿A). This type of disposition is also useful for automated transcoding services, where one machine converts text to synthetic speech (text-to-speech) and a different machine performs voice recognition (speech-to-text).

The scenario just described involves four different sessions; A-T1, T1-B, B-T2 and T2-A. Figure 6 shows the call flow where A invokes T1 and T2.

(1) INVITE SDP AT1

```
m=text 20000 RTP/AVP 96
c=IN IP4 A.domain.com
a=rtpmap:96 t140/1000
a=sendonly
a=source:1
m=audio 20000 RTP/AVP 0
c=IN IP4 0.0.0.0
a=recvonly
a=sink:1
```

(2) INVITE SDP AT2

```
m=text 20002 RTP/AVP 96
c=IN IP4 A.domain.com
a=rtpmap:96 t140/1000
a=recvonly
a=sink:1
```

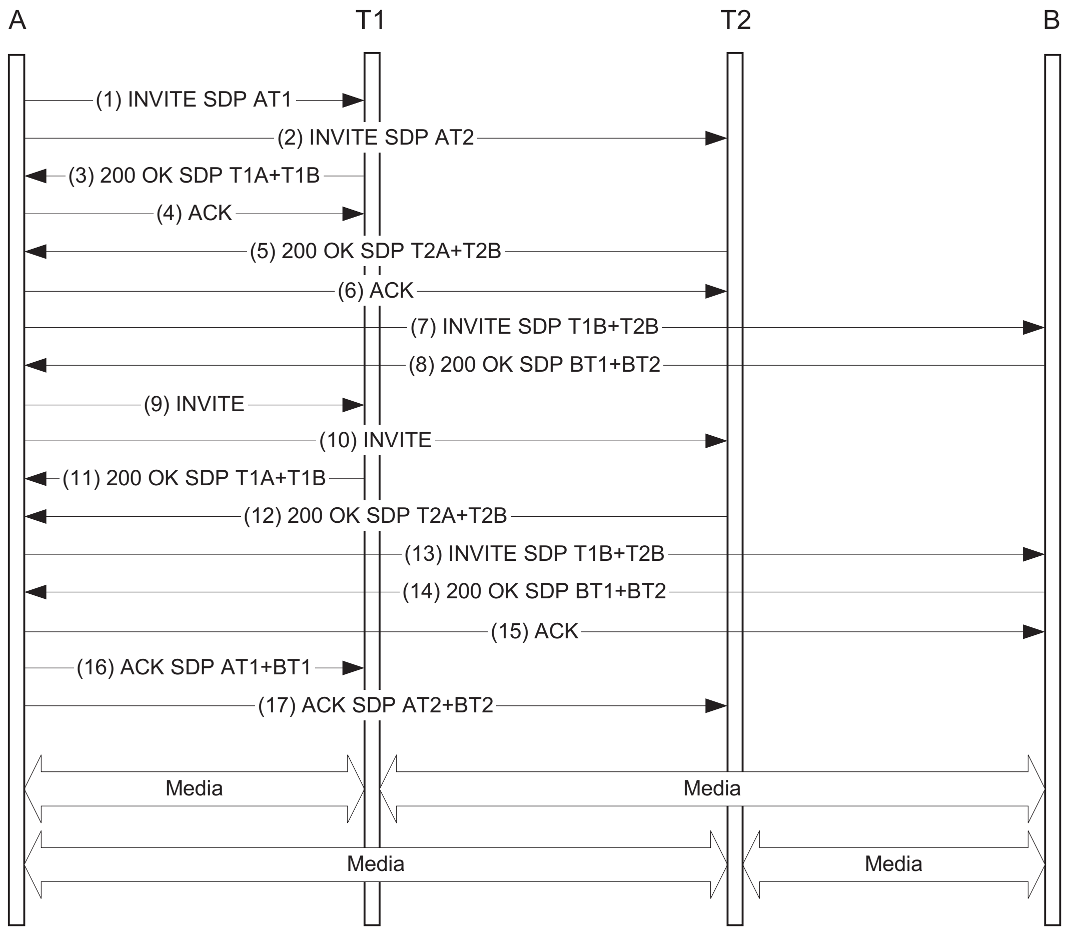

Figure 6: Transcoding services in parallel

```
m=audio 20000 RTP/AVP 0
c=IN IP4 0.0.0.0
a=sendonly
a=source:1
```

(3) 200 OK SDP T1A+T1B

```
m=text 30000 RTP/AVP 96
c=IN IP4 T1.domain.com
a=rtpmap:96 t140/1000
a=recvonly
a=source:1
m=audio 30002 RTP/AVP 0
c=IN IP4 T1.domain.com
a=sendonly
a=sink:1
```

(5) 200 OK SDP T2A+T2B

```
m=text 40000 RTP/AVP 96
c=IN IP4 T2.domain.com
a=rtpmap:96 t140/1000
a=sendonly
a=sink:1
m=audio 40002 RTP/AVP 0
c=IN IP4 T2.domain.com
a=recvonly
a=source:1
```

(7) INVITE SDP T1B+T2B

```
m=audio 30002 RTP/AVP 0
c=IN IP4 T1.domain.com
a=sendonly
m=audio 40002 RTP/AVP 0
c=IN IP4 T2.domain.com
a=recvonly
```

(8) 200 OK SDP BT1+BT2

```
m=audio 50000 RTP/AVP 0
c=IN IP4 B.domain.com
a=recvonly
m=audio 50002 RTP/AVP 0
c=IN IP4 B.domain.com
a=sendonly
```

(11) 200 OK SDP T1A+T1B

```
m=text 30000 RTP/AVP 96
c=IN IP4 T1.domain.com
a=rtpmap:96 t140/1000
a=recvonly
a=source:1
m=audio 30002 RTP/AVP 0
c=IN IP4 T1.domain.com
a=sendonly
a=sink:1
```

(12) 200 OK SDP T2A+T2B

```
m=text 40000 RTP/AVP 96
c=IN IP4 T2.domain.com
a=rtpmap:96 t140/1000
a=sendonly
a=sink:1
m=audio 40002 RTP/AVP 0
c=IN IP4 T2.domain.com
a=recvonly
a=source:1
```

Since T1 have returned the same SDP in (11) as in (3) and T2 has returned the same SDP in (12) as in (5), messages (13), (14) and (15) can be skipped.

(16) ACK SDP AT1+BT1

```
m=text 20000 RTP/AVP 96
c=IN IP4 A.domain.com
a=rtpmap:96 t140/1000
a=sendonly
a=source:1
m=audio 50000 RTP/AVP 0
c=IN IP4 B.domain.com
a=recvonly
a=sink:1
```

(17) ACK SDP AT2+BT2

```
m=text 20002 RTP/AVP 96
c=IN IP4 A.domain.com
a=rtpmap:96 t140/1000
a=recvonly
a=sink:1
m=audio 50002 RTP/AVP 0
```

```
        c=IN IP4 B.domain.com
        a=sendonly
        a=source:1
```

Four media streams have been established at this point:

1. Text from A to T1.domain.com:30000
2. Audio from T1 to B.domain.com:50000
3. Audio from B to T2.domain.com:40002
4. Text from T2 to A.domain.com:20002

Note that B, the user agent server, needs to support two media streams; one sendonly and the other recvonly. At present, some user agents, although they support a single sendrecv media stream, they do not support a different media line per direction. Implementers are encouraged to build support for this feature.

3.5 Transcoding services in serial

In a distributed environment, a complex transcoding service (e.g., English text to Spanish speech) is often provided by several servers. For example, one server performs English text to Spanish text translation, and its output is feed into a server that performs text-to-speech conversion. The flow in Figure 7 shows how A invokes T1 and T2.

4 Authors' Addresses

Gonzalo Camarillo
Ericsson
Advanced Signalling Research Lab.
FIN-02420 Jorvas
Finland
electronic mail: Gonzalo.Camarillo@ericsson.com

Eric W. Burger
SnowShore Networks, Inc.
Chelmsford, MA
USA
electronic mail: eburger@snowshore.com

Henning Schulzrinne
Dept. of Computer Science
Columbia University 1214 Amsterdam Avenue, MC 0401
New York, NY 10027
USA
electronic mail: schulzrinne@cs.columbia.edu

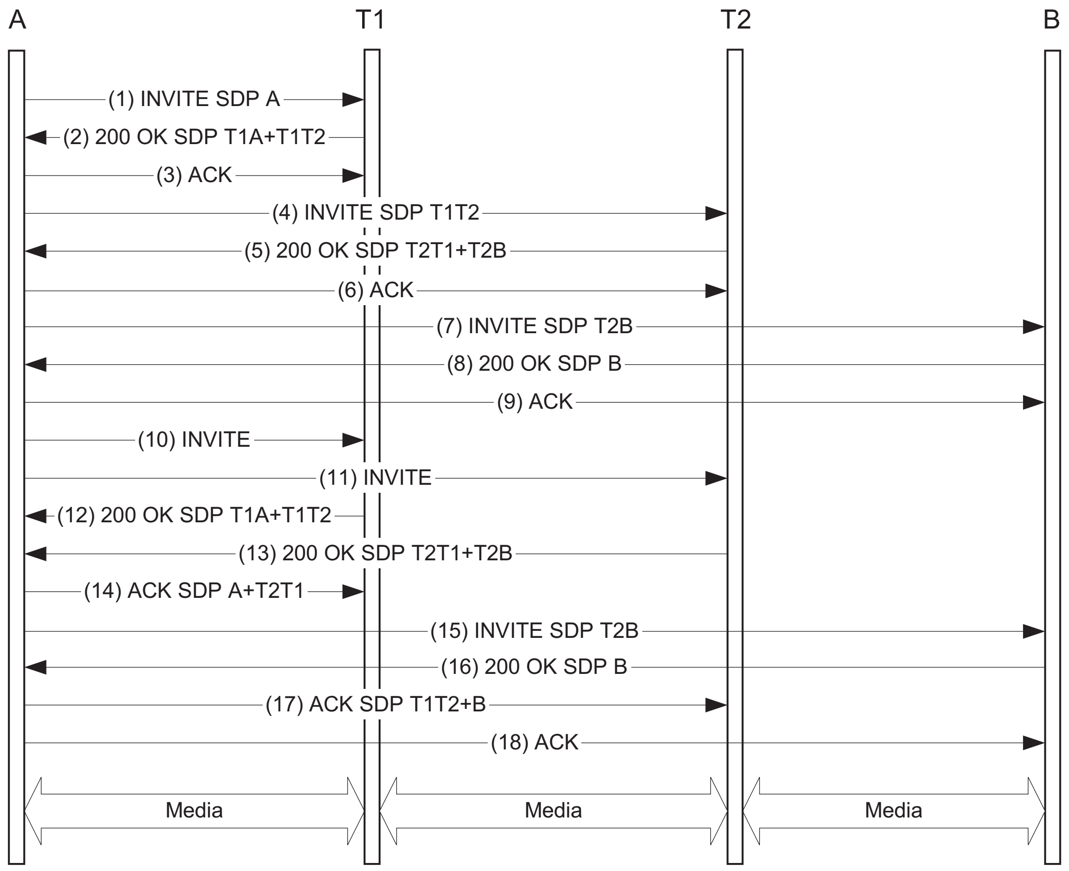

Figure 7: Transcoding services in serial

Arnoud van Wijk
Ericsson EuroLab Netherlands BV
P.O. Box 8
5120 AA Rijen
The Netherlands
electronic mail: Arnoud.van.Wijk@eln.ericsson.se

References

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: session initiation protocol," RFC 3261, Internet Engineering Task Force, June 2002.

[2] N. Charlton, M. Gasson, G. Gybels, M. Spanner, and A. van Wijk, "User requirements for the session initiation protocol (SIP) in support of deaf, hard of hearing and speech-impaired individuals," RFC 3351, Internet Engineering Task Force, Aug. 2002.

[3] J. Rosenberg, J. Peterson, H. Schulzrinne, and G. Camarillo, "Best current practices for third party call control in the session initiation protocol," Internet Draft, Internet Engineering Task Force, June 2002. Work in progress.

[4] M. Handley and V. Jacobson, "SDP: session description protocol," RFC 2327, Internet Engineering Task Force, Apr. 1998.

[5] G. Camarillo, H. Schulzrinne, and E. Burger, "The source and sink attributes for the session description protocol," Internet Draft, Internet Engineering Task Force, Sept. 2002. Work in progress.

[6] G. Camarillo, J. Holler, G. Eriksson, and H. Schulzrinne, "Grouping of m lines in SDP," Internet Draft, Internet Engineering Task Force, Feb. 2002. Work in progress.

Full Copyright Statement

Copyright (c) The Internet Society (2002). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS

ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.