SIP Endpoint SDK Overview

SIP Endpoint SDK 8.5.0NET

1/4/2022

Table of Contents

Overview

The SIP Endpoint SDK enables you to build a SIP endpoint that can seamlessly connect agent desktop applications with the Genesys SIP Server in order to handle audio and video calls. This overview will help you understand its design goals and its architecture.

Design Goals

Many SIP softphones that are currently available require an agent to interact with their own separate user interface in addition to that of the agent desktop application. For instance, the agent might have to use the SIP phone interface to answer a call, while other actions — such as holding or releasing the call — would have to be done via agent desktop interface.

In contrast to that, the SIP Endpoint SDK is designed to be integrated into an agent desktop so the agent can use a single user interface to control calls. Genesys recommends that this be done in a way that leaves actual control in the hands of a T-Lib–based agent desktop application, which has a fuller feature set and is also fully supported by Genesys.

The SIP Endpoint SDK is also designed to integrate with the Genesys SIP Server. It supports the SIP, SDP, and RTP/RTCP protocols.

In addition to these principal design goals, the SIP Endpoint SDK supports the following features:

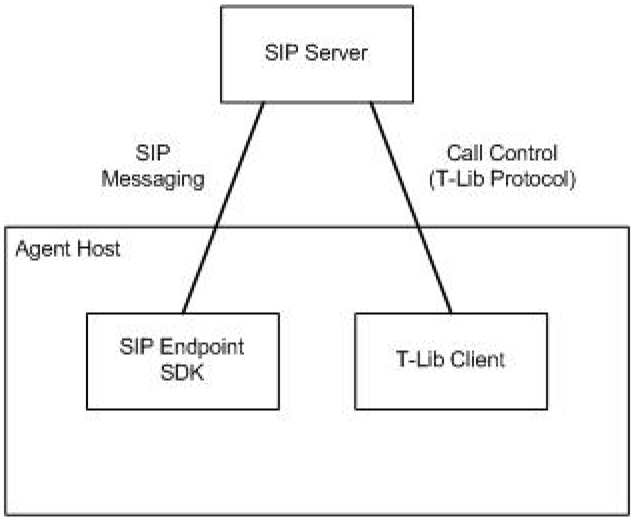
- Traditional call control functionality such as:
    - Establishing inbound and outbound calls
    - Hold and retrieve
    - Transfers and conference calls (Note that SIP Endpoint SDK does not handle transfers and conference calls directly. You must use third-party call control in order to enable these features.)
- Multi-line call handling

Note: Multi-line call handling is available up to the limitations on the number of simultaneous connections supported by the licensing available to Genesys at this time.

Architecture

The SIP Endpoint SDK is a software component that resides on the agent's computer. It uses the SIP protocol to communicate with the Genesys SIP Server and supports both thin (or 3-tier) and thick (or rich) T-Lib clients.

In SIP messaging terms, the SIP Endpoint is a user agent. The contact center agent's computer should also host a Genesys T-Lib client that works with SIP Server and provides Genesys agent-related functionality, as shown below.



As you can see, the SIP Endpoint SDK handles the SIP messaging, and the T-Lib Client uses the T-Lib protocol to handle call control.

SDK Components

The components of the SIP Endpoint SDK are described on the following page:

- SIP Endpoint SDK for NET Components

Supported Codecs

SIP Endpoint SDK 8.5.0 supports the following codecs:

- G722/16000 (G.722)
- ILBC/8000 (iLBC — internet Low Bitrate Codec)
- ISAC/16000 (iSAC/16kHz — internet Speech Audio Codec)
- ISAC/32000 (iSAC/32kHz)
- OPUS/48000/2
- PCMA/8000 (G.711/A-law)
- PCMU/8000 (G.711/mu-law)
- VP8 video

Important

Forward error correction is supported using ULPFEC (RFC 5109). SIP Endpoint SDK automatically adds this support as long as `ulpfec` is included in the codec list in the configuration file.

Supported RFCs

SIP Endpoint SDK 8.5.0 partially or fully supports the following RFCs:

Section	Name	Description
Media		
	RFC 1889	RTP: A Transport Protocol for Real-Time Applications
	RFC 2327	SDP: Session Description Protocol
	RFC 2833	RTP Payload for DTMF Digits, Telephony Tones and Telephony Signals
	RFC 3264	An Offer/Answer Model with the Session Description Protocol (SDP)
	RFC 3550	RTP: A Transport Protocol for Real-Time Applications [replaces RFC 1889]
	RFC 3951	Internet Low Bit Rate Codec (iLBC)
	RFC 3952	Real-time Transport Protocol (RTP) Payload Format for internet Low Bit Rate Codec (iLBC) Speech
	RFC 5109	RTP Payload Format for Generic Forward Error Correction
Network		
	RFC 1035	Domain names—implementation and specification
	RFC 2327	SDP: Session Description Protocol
SIP		
	draft-ietf-sipping-cc-transfer	Session Initiation Protocol Call Control—Transfer draft-ietf-sipping-cc-transfer-12
	RFC 2617	HTTP Authentication: Basic and Digest Access Authentication (for SIP)
	RFC 2976	The SIP INFO Method
	RFC 3261	SIP: Session Initiation Protocol
	RFC 3265	Session Initiation Protocol (SIP): Specific Event Notification
	RFC 3420	Internet Media Type message/sipfrag
	RFC 3515	The Session Initiation Protocol (SIP) Refer Method
	RFC 3891	The Session Initiation Protocol

Section	Name	Description
		(SIP) "Replaces" Header
	RFC 3892	The Session Initiation Protocol (SIP) Referred-By Mechanism

Working with the SIP Endpoint SDK for .NET

The SIP Endpoint SDK for .NET distribution includes the following files, which you can use "as is" in your custom applications:

- Genesyslab.Sip.Endpoint.dll
- Genesyslab.Sip.Endpoint.Provider.Genesys.dll

These files are located in the `\Bin` directory at the root level of the SIP Endpoint SDK directory.

The SIP Endpoint SDK also depends on the following Genesys and third-party libraries, which should be present in the working directory:

- intl.dll
- libgio-2.0-0.dll
- libglib-2.0-0.dll
- libgmodule-2.0-0.dll
- libgobject-2.0-0.dll
- libgthread-2.0-0.dll
- libnice.dll
- zlib1.dll
- Genesyslab.Core.dll
- Genesyslab.Platform.Commons.Collections.dll
- Genesyslab.Platform.Commons.Connection.dll
- Genesyslab.Platform.Commons.dll
- Genesyslab.Platform.Commons.Protocols.dll
- Genesyslab.Platform.Logging.dll
- Genesyslab.Platform.Management.Protocols.dll
- Genesyslab.Sip.Endpoint.dll
- Genesyslab.Sip.Endpoint.Provider.Genesys.dll

Learning More

To continue learning about the SIP Endpoint SDK, we recommend you read the pages describing SIP-

Based Third-Party Call Control to understand messaging patterns. After that, you should be ready to use the Deployment Guide to install the SDK on your system.

Once you have installed the SDK, you may want to check the following pages from the Developer's Guide:

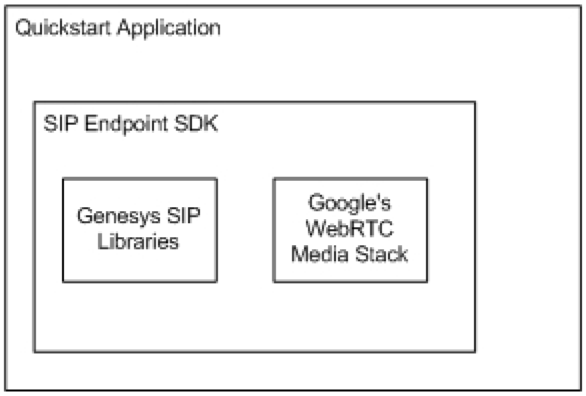
- SIP Endpoint SDK Configuration for .NET
- SIP Endpoint SDK Disaster Recovery and Geo-Redundancy

SIP Endpoint SDK for .NET Components

The SIP Endpoint SDK distribution consists of the following main components:

- The SDK, which provides all of the SIP-related functionality. The SDK also provides an interface that you can use to integrate it into different GUI-based applications.
- A sample QuickStart application which is built on the SIP Endpoint SDK.

The SDK itself runs on top of the Genesys SIP libraries and Google's WebRTC media stack, while the QuickStart application runs on top of the SDK, as shown here:



SIP-Based Third-Party Call Control

This section provides sequence diagrams and descriptions of the call scenarios supported by SIP Server and the SIP Endpoint SDK, which are achieved using the T-Lib API. It also demonstrates how the T-Lib API is mapped into SIP messaging.

Note that the SIP Endpoint SDK supports the talk/hold NOTIFY extension.

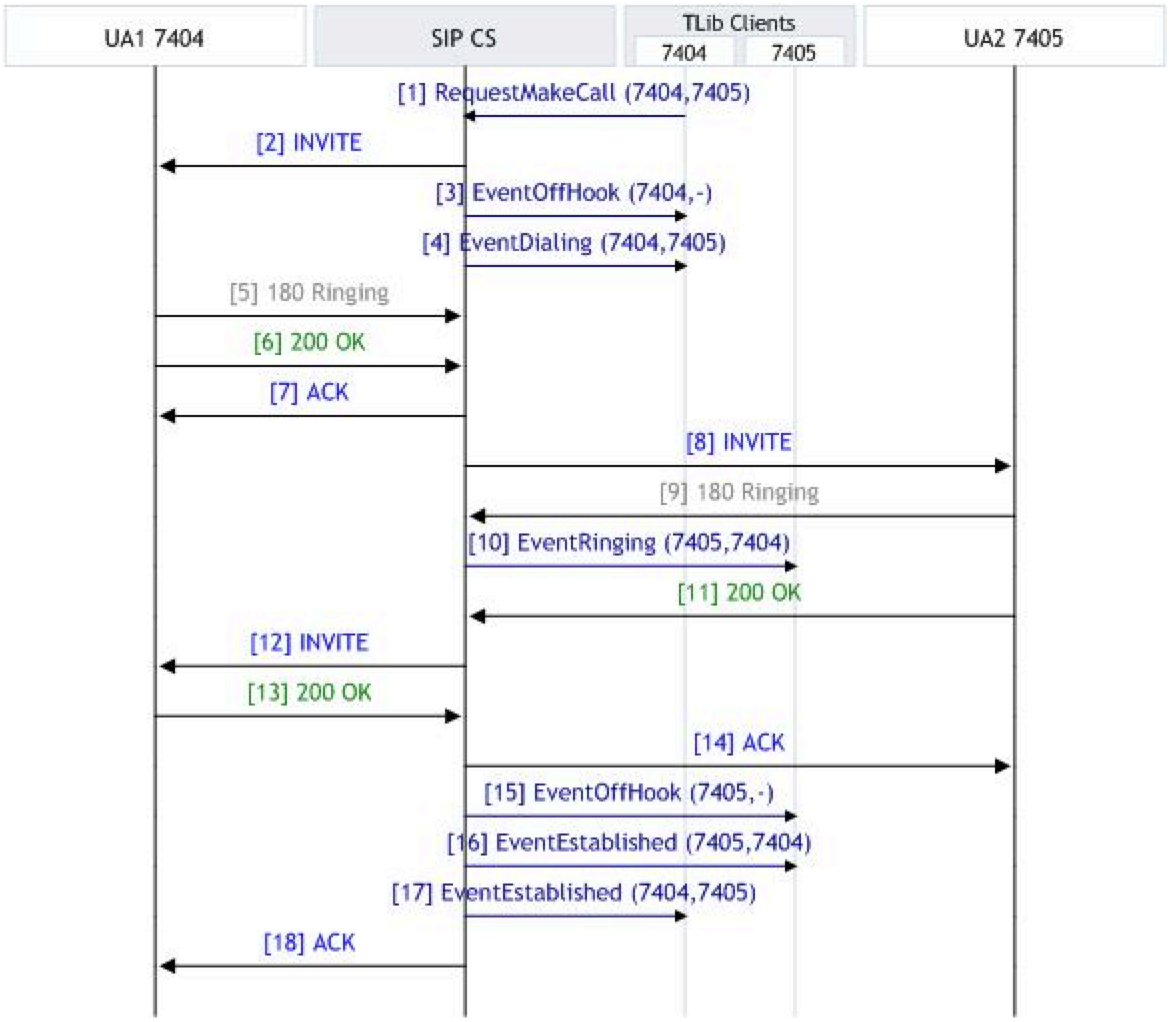
T-Lib-Initiated MakeCall re-INVITE

Description

UA1 calls UA2. If the UA1 endpoint does not support the REFER method, then the re-INVITE mechanism could be used. The main disadvantages of re-INVITE are:

- The phone may not reflect the real call progress unless early media is activated between both endpoints.
- Re-INVITE makes it next to impossible for third-party switches to track call topology and accumulate accurate call detail records (CDR).

Diagram


UA1 7404
SIP CS
TLib Clients
7404
7405
UA2 7405
[1] RequestMakeCall (7404,7405)
[2] INVITE
[3] EventOffHook (7404,-)
[4] EventDialing (7404,7405)
[5] 180 Ringing
[6] 200 OK
[7] ACK
[8] INVITE
[9] 180 Ringing
[10] EventRinging (7405,7404)
[11] 200 OK
[12] INVITE
[13] 200 OK
[14] ACK
[15] EventOffHook (7405,-)
[16] EventEstablished (7405,7404)
[17] EventEstablished (7404,7405)
[18] ACK

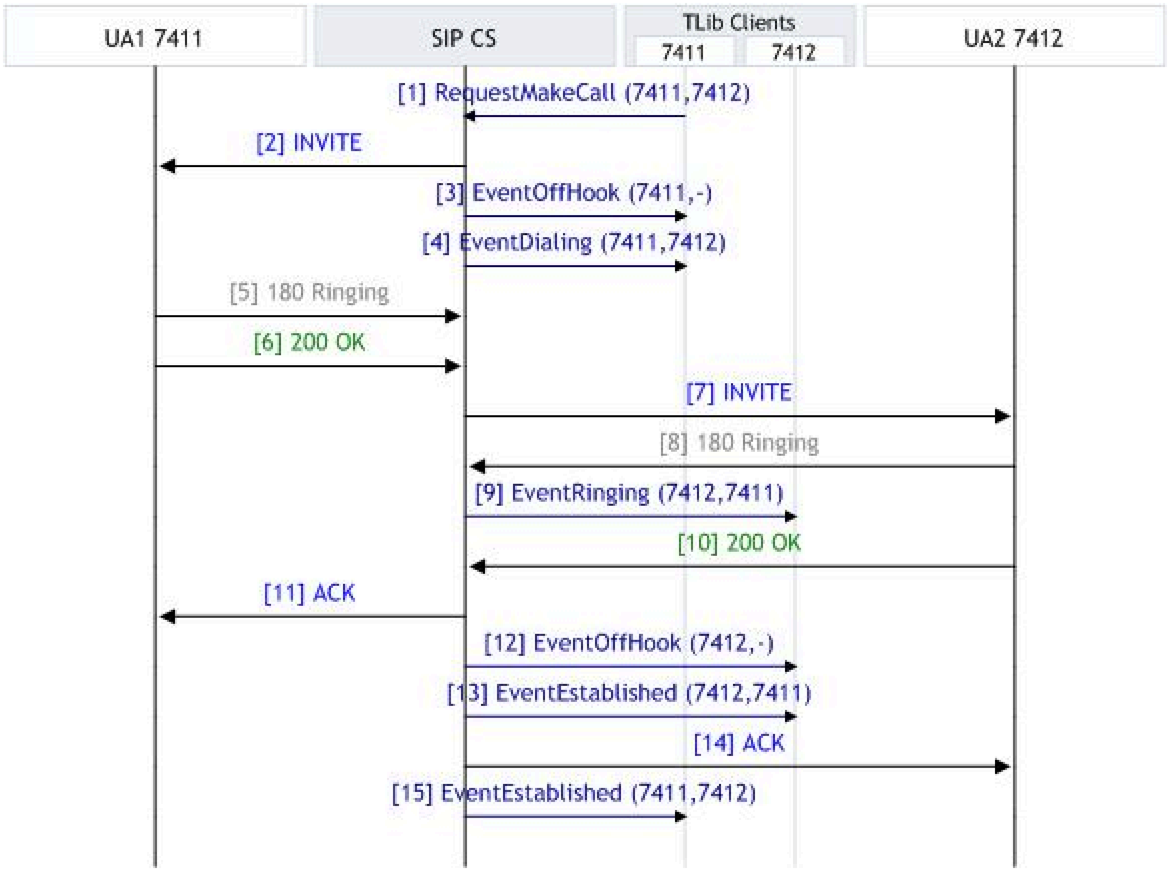
T-Lib-Initiated MakeCall INVITE

Description

RFC 3725 is a configuration flag that can be set to 2 (the default, which causes the T-Lib-Initiated MakeCall re-INVITE call scenario to be used) or 1. When the parameter is set to 1, the call flow will be as given below. Notice in this case that according to the RFC (best current practices for third-party call control) UA1 does not get an ACK until UA2 has responded with `200 OK`. In this case, SIP Server does not send a re-INVITE to UA1.

Note: Usually, the re-INVITE call flow is the preferred use case. This is the default behavior of SIP Server, unless the RFC3725 parameter is set to 1.

Diagram

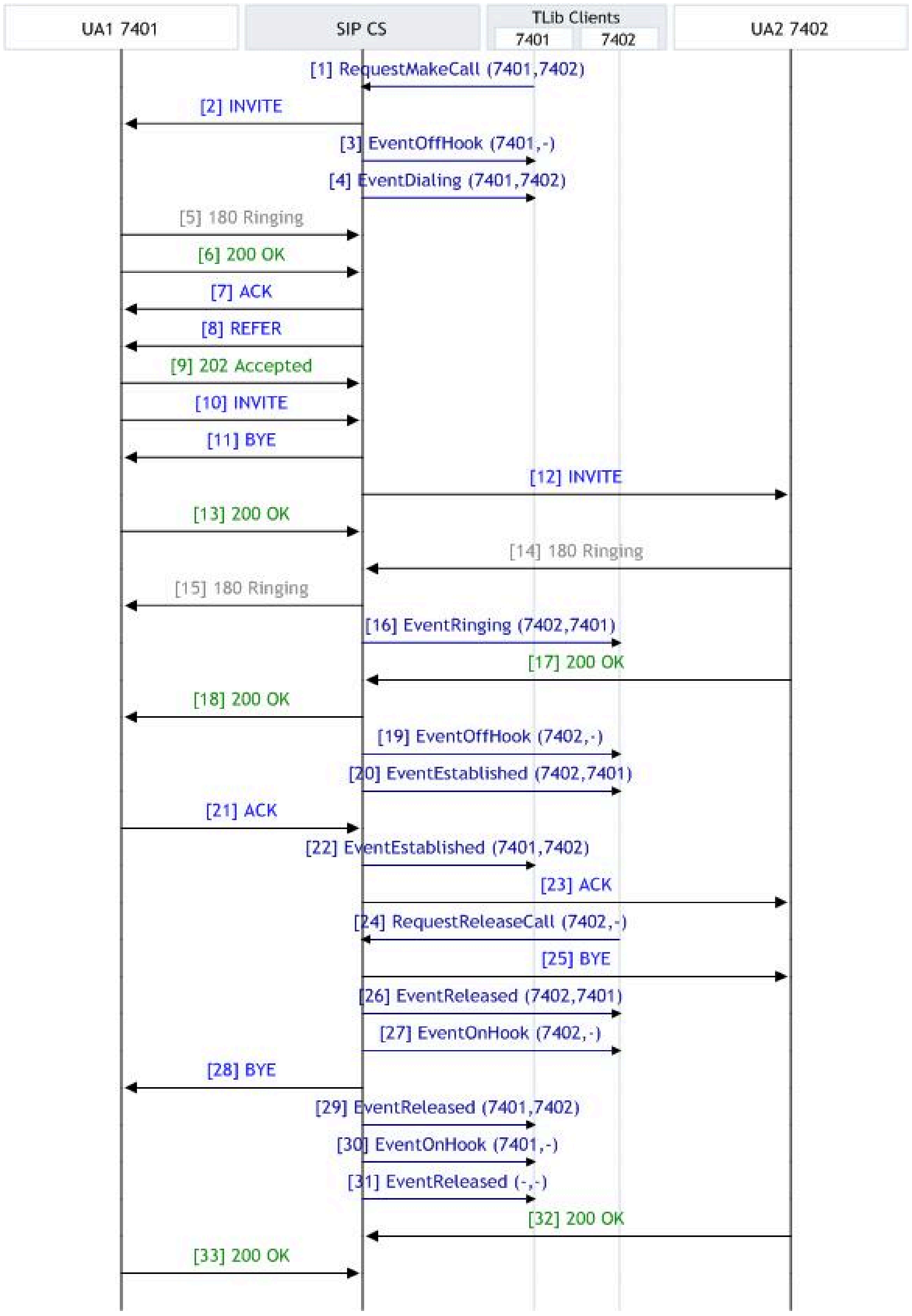

UA1 7411
SIP CS
TLib Clients
7411
7412
UA2 7412
[1] RequestMakeCall (7411,7412)
[2] INVITE
[3] EventOffHook (7411,-)
[4] EventDialing (7411,7412)
[5] 180 Ringing
[6] 200 OK
[7] INVITE
[8] 180 Ringing
[9] EventRinging (7412,7411)
[10] 200 OK
[11] ACK
[12] EventOffHook (7412,-)
[13] EventEstablished (7412,7411)
[14] ACK
[15] EventEstablished (7411,7412)

T-Lib Termination TReleaseCall

Description

The T-Lib client initiates the termination of the call by invoking TReleaseCall from the agent desktop at UA2. SIP Server terminates the existing call from UA1 to UA2 by sending two BYE requests.

Diagram


UA1 7401
SIP CS
TLib Clients
7401
7402
UA2 7402
[1] RequestMakeCall (7401,7402)
[2] INVITE
[3] EventOffHook (7401,-)
[4] EventDialing (7401,7402)
[5] 180 Ringing
[6] 200 OK
[7] ACK
[8] REFER
[9] 202 Accepted
[10] INVITE
[11] BYE
[12] INVITE
[13] 200 OK
[14] 180 Ringing
[15] 180 Ringing
[16] EventRinging (7402,7401)
[17] 200 OK
[18] 200 OK
[19] EventOffHook (7402,-)
[20] EventEstablished (7402,7401)
[21] ACK
[22] EventEstablished (7401,7402)
[23] ACK
[24] RequestReleaseCall (7402,-)
[25] BYE
[26] EventReleased (7402,7401)
[27] EventOnHook (7402,-)
[28] BYE
[29] EventReleased (7401,7402)
[30] EventOnHook (7401,-)
[31] EventReleased (-,-)
[32] 200 OK
[33] 200 OK

T-Lib-Initiated Hold-Retrieve

Description

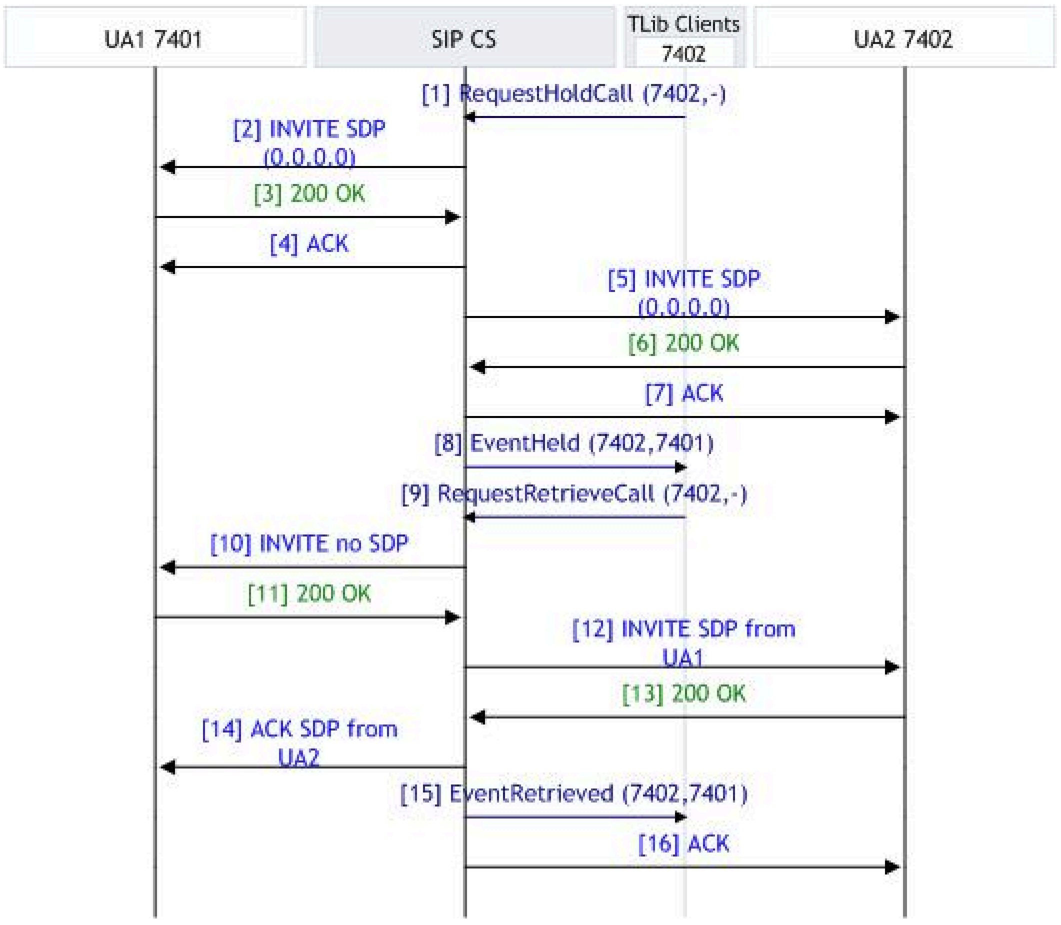
SIP Server's Hold operation can be configured to play music while the customer is placed on hold by an agent. The music-on-hold configuration is global — if it is enabled, all held parties hear music; if it is not enabled, all held parties hear silence.

Hold-with-silence only involves stopping the RTP streams. The Retrieve operation resumes the RTP streams.

There are multiple methods for stopping the RTP streams. SIP Server uses IP address 0.0.0.0 in the `c=` attribute as the only universally accepted method.

Because some endpoints may change the RTP port address when trying to retrieve a call, the re-INVITE with delayed SDP offer-answer negotiation shown in the following diagram is necessary.

Diagram


UA1 7401
SIP CS
TLib Clients 7402
UA2 7402
[1] RequestHoldCall (7402,-)
[2] INVITE SDP (0.0.0.0)
[3] 200 OK
[4] ACK
[5] INVITE SDP (0.0.0.0)
[6] 200 OK
[7] ACK
[8] EventHeld (7402,7401)
[9] RequestRetrieveCall (7402,-)
[10] INVITE no SDP
[11] 200 OK
[12] INVITE SDP from UA1
[13] 200 OK
[14] ACK SDP from UA2
[15] EventRetrieved (7402,7401)
[16] ACK

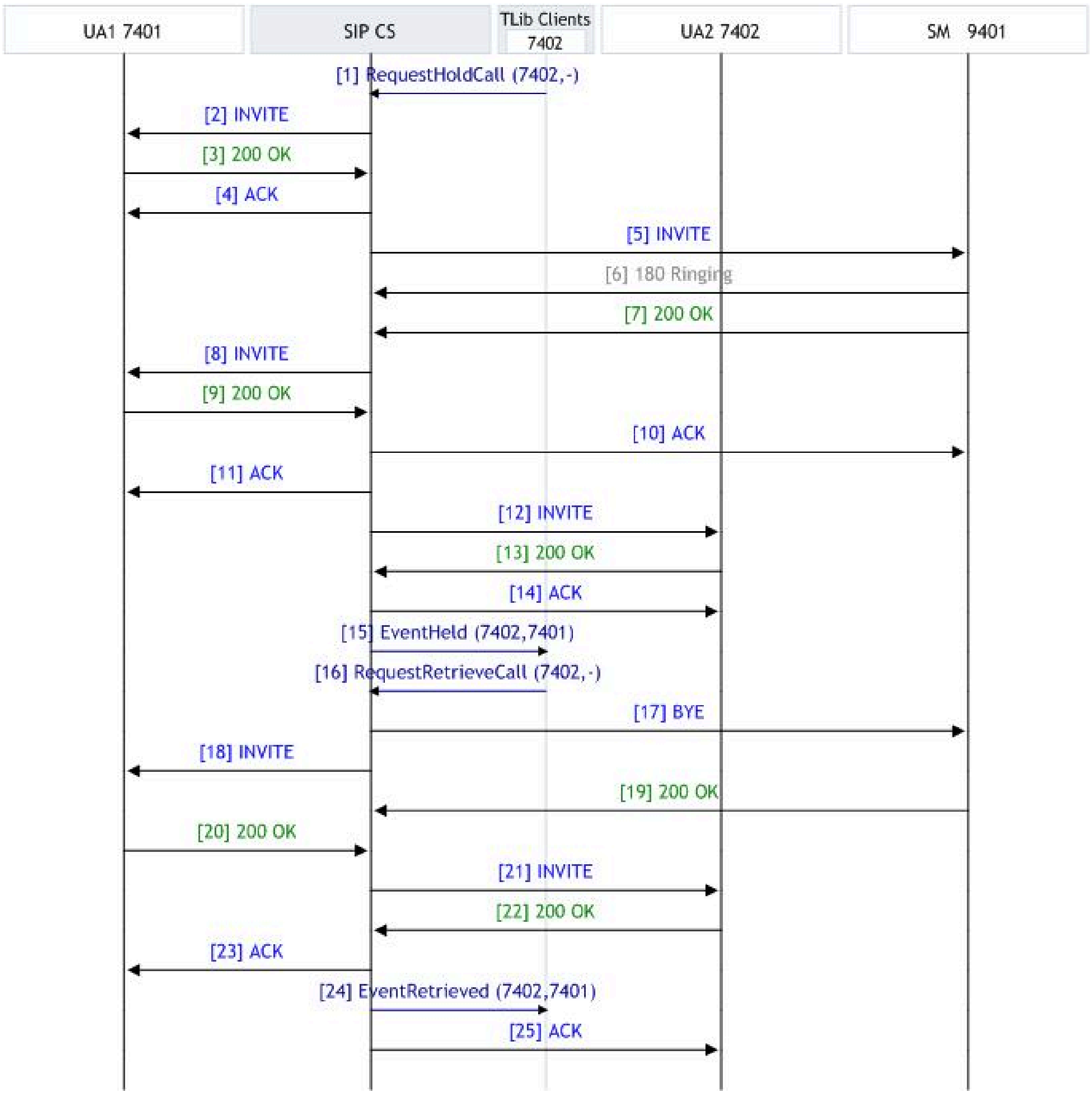
T-Lib-Initiated Hold-Retrieve Music on Hold

Description

The Music-on-Hold (MOH) Server is a regular SIP UA that streams music in an RTP stream. Stream Manager is used to provide Music On Hold.

In the scenario illustrated below, the agent at UA1 places UA2 on hold by invoking the T-Lib request THoldCall. UA2 will hear music while on hold; UA1 will hear silence.

Diagram


UA1 7401
SIP CS
TLib Clients 7402
UA2 7402
SM 9401
[1] RequestHoldCall (7402,-)
[2] INVITE
[3] 200 OK
[4] ACK
[5] INVITE
[6] 180 Ringing
[7] 200 OK
[8] INVITE
[9] 200 OK
[10] ACK
[11] ACK
[12] INVITE
[13] 200 OK
[14] ACK
[15] EventHeld (7402,7401)
[16] RequestRetrieveCall (7402,-)
[17] BYE
[18] INVITE
[19] 200 OK
[20] 200 OK
[21] INVITE
[22] 200 OK
[23] ACK
[24] EventRetrieved (7402,7401)
[25] ACK

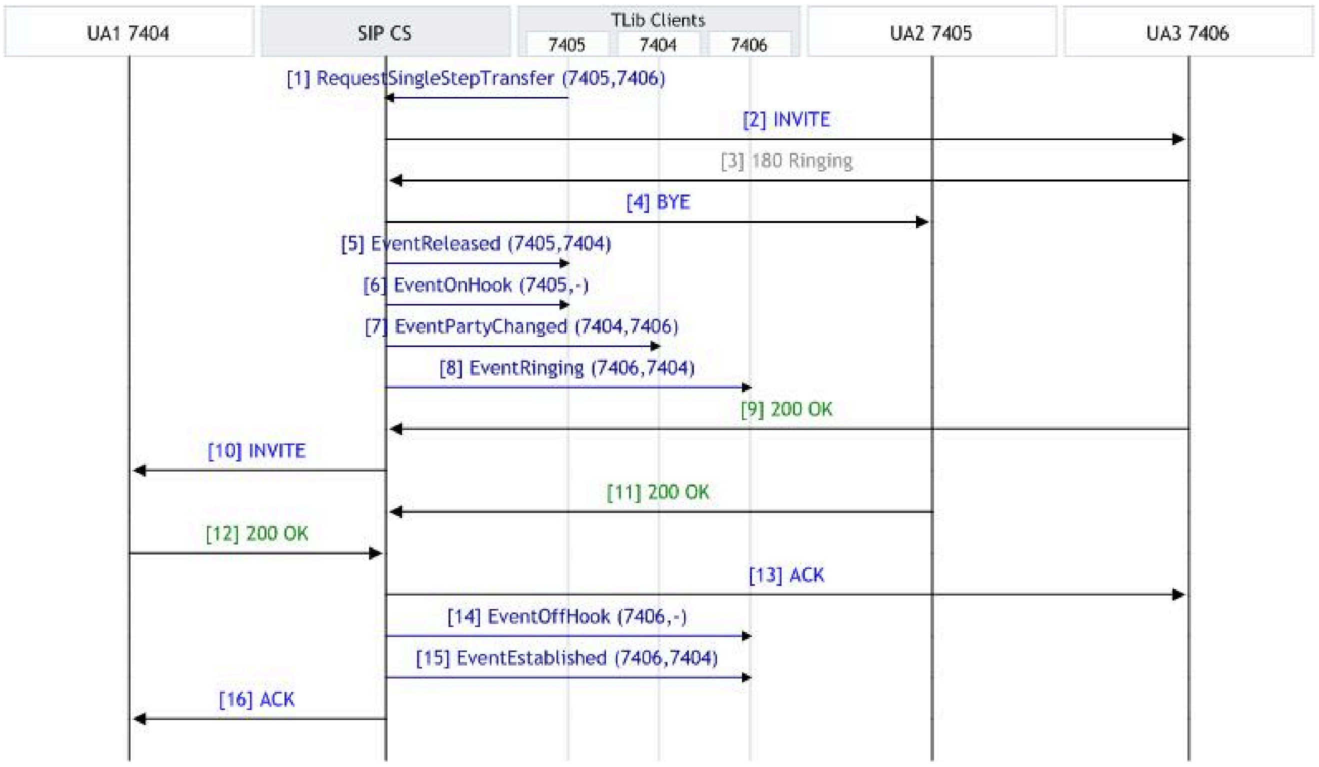
T-Lib-Initiated SingleStepTransfer re-INVITE

Description

UA1 calls UA2, and the agent at UA2 invokes a T-Lib request to transfer the call from UA2 to UA3.

For the advantages and disadvantages of the re-INVITE mechanism involved in this scenario, see the description in the section on T-Lib-Initiated MakeCall re-INVITE.

Diagram


UA1 7404
SIP CS
TLib Clients
7405
7404
7406
UA2 7405
UA3 7406
[1] RequestSingleStepTransfer (7405,7406)
[2] INVITE
[3] 180 Ringing
[4] BYE
[5] EventReleased (7405,7404)
[6] EventOnHook (7405,-)
[7] EventPartyChanged (7404,7406)
[8] EventRinging (7406,7404)
[9] 200 OK
[10] INVITE
[11] 200 OK
[12] 200 OK
[13] ACK
[14] EventOffHook (7406,-)
[15] EventEstablished (7406,7404)
[16] ACK

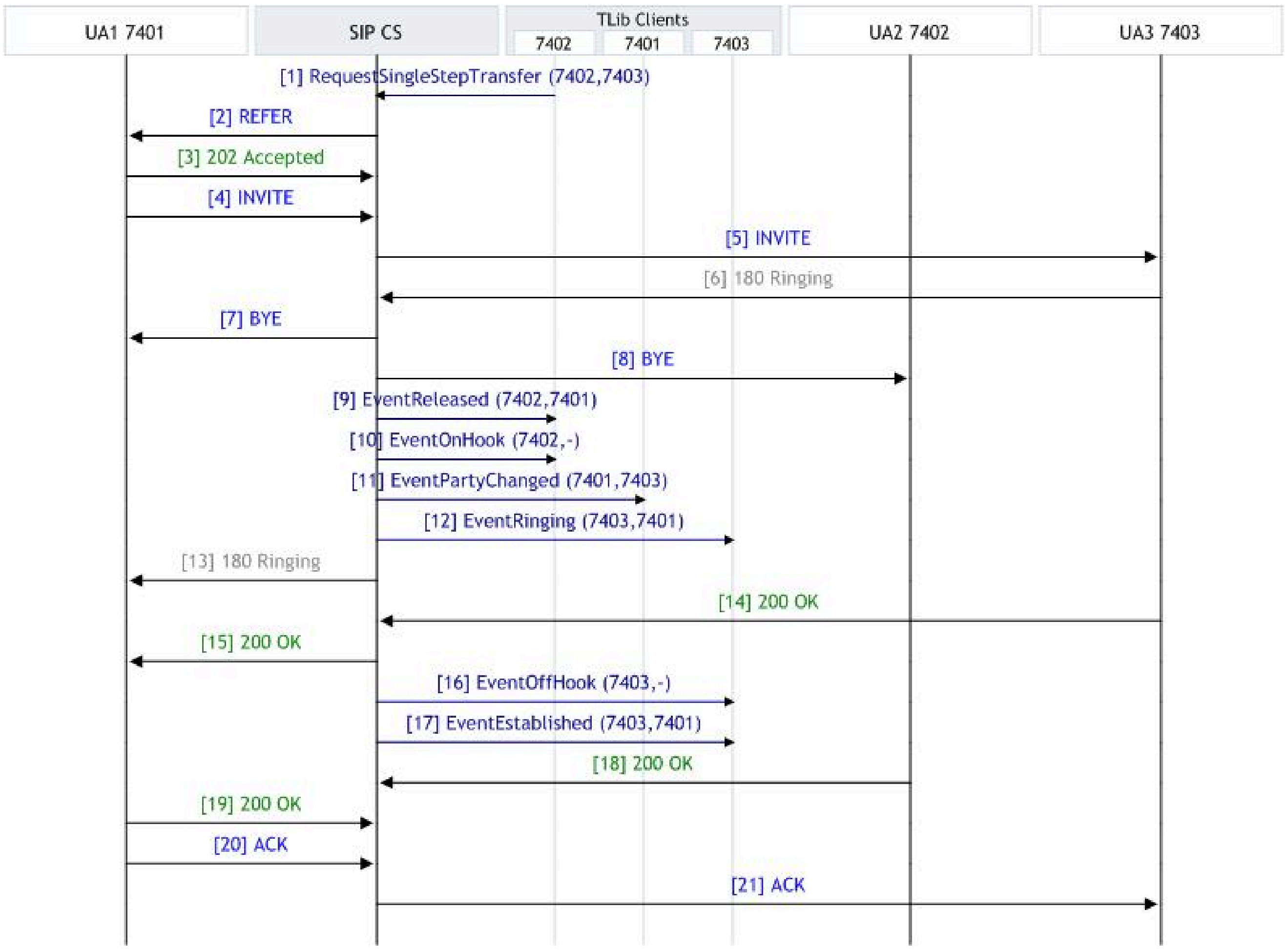
T-Lib-Initiated SingleStepTransfer REFER

Description

UA1 calls UA2, and the T-Lib request transfers the call from UA2 to UA3.

This scenario is equivalent to a single-step transfer using the REFER method executed from UA2, except that in this case, the REFER message originates on SIP Server, not on UA2. Also, the "202 Accepted" message comes back to SIP Server, not UA2, and SIP Server also disconnects the call between UA1 and UA2 using a third-party call control release.

Diagram

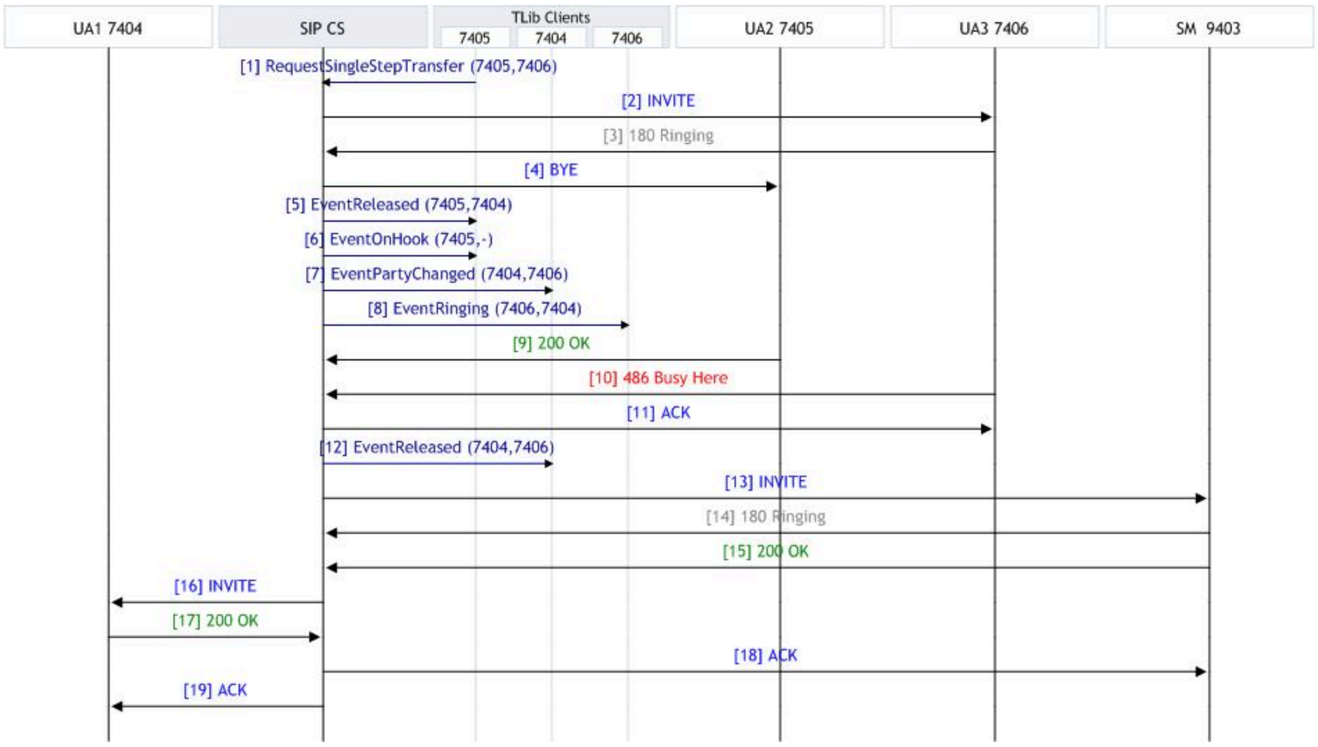

UA1 7401
SIP CS
TLib Clients
7402
7401
7403
UA2 7402
UA3 7403
[1] RequestSingleStepTransfer (7402,7403)
[2] REFER
[3] 202 Accepted
[4] INVITE
[5] INVITE
[6] 180 Ringing
[7] BYE
[8] BYE
[9] EventReleased (7402,7401)
[10] EventOnHook (7402,-)
[11] EventPartyChanged (7401,7403)
[12] EventRinging (7403,7401)
[13] 180 Ringing
[14] 200 OK
[15] 200 OK
[16] EventOffHook (7403,-)
[17] EventEstablished (7403,7401)
[18] 200 OK
[19] 200 OK
[20] ACK
[21] ACK

T-Lib-Initiated SingleStepTransfer User Busy

Description

If a single-step transfer is attempted and UA3 returns a busy signal or a no-answer event and rejects the attempted transfer, UA2 may or may not be available to pick up the dropped call. Note that due to the nature of single-step transfers, a call can be dropped in these cases. In addition to that, there is no guarantee that UA2 will still be available if UA3 reports busy or does not answer within a reasonable time. Thus, if your contact center requires that calls not be dropped, you should use consult transfer, not single-step transfer.

Diagram



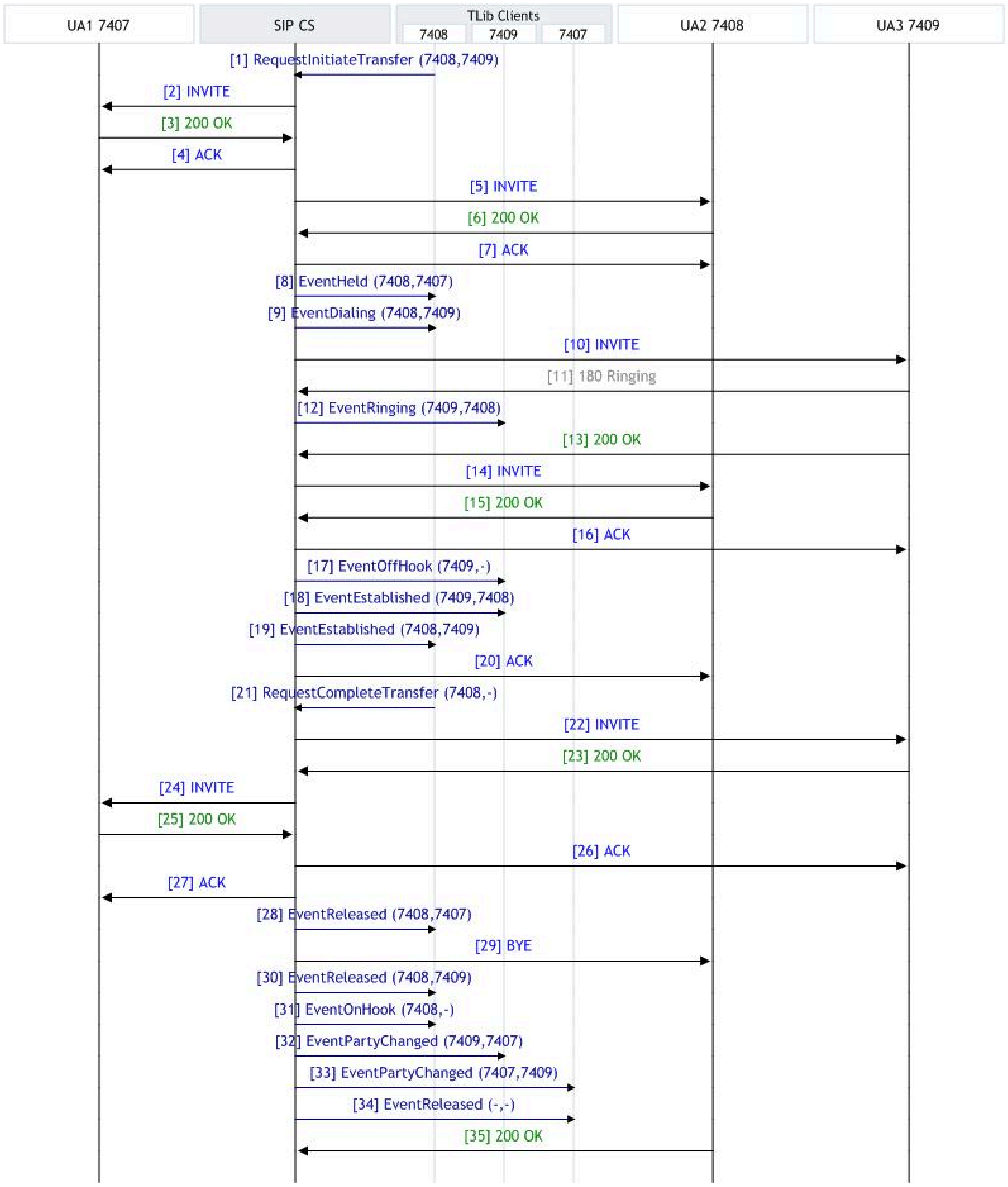
T-Lib-Initiated Consult Call Single EndPoint

Description

There are some cases when it is not possible to use the normal means of initiating a consult call by setting the dialog of the main call on hold and creating a new dialog for the consult call. One such case involves a transfer initiated by a remote agent with a regular (non-VoIP) phone behind the media-gateway. In that case, a consult call can be initiated by "borrowing" the SIP Dialog from the main call and re-using it for the consult call.

The diagram below describes this kind of case. To provide Ring-Back to the agent during the progress of the call, the SIP Dialog is invited to Stream Manager, which will play a ring-back tone. If the consult call is finished by a TCompleteTransfer request (as shown in the diagram), that dialog will be terminated. If the agent decides to reconnect the call, the SIP Dialog will be returned to the main call and retrieved.

Diagram


UA1 7407
SIP CS
TLib Clients
7408
7409
7407
UA2 7408
UA3 7409
[1] RequestInitiateTransfer (7408,7409)
[2] INVITE
[3] 200 OK
[4] ACK
[5] INVITE
[6] 200 OK
[7] ACK
[8] EventHeld (7408,7407)
[9] EventDialing (7408,7409)
[10] INVITE
[11] 180 Ringing
[12] EventRinging (7409,7408)
[13] 200 OK
[14] INVITE
[15] 200 OK
[16] ACK
[17] EventOffHook (7409,-)
[18] EventEstablished (7409,7408)
[19] EventEstablished (7408,7409)
[20] ACK
[21] RequestCompleteTransfer (7408,-)
[22] INVITE
[23] 200 OK
[24] INVITE
[25] 200 OK
[26] ACK
[27] ACK
[28] EventReleased (7408,7407)
[29] BYE
[30] EventReleased (7408,7409)
[31] EventOnHook (7408,-)
[32] EventPartyChanged (7409,7407)
[33] EventPartyChanged (7407,7409)
[34] EventReleased (-,-)
[35] 200 OK

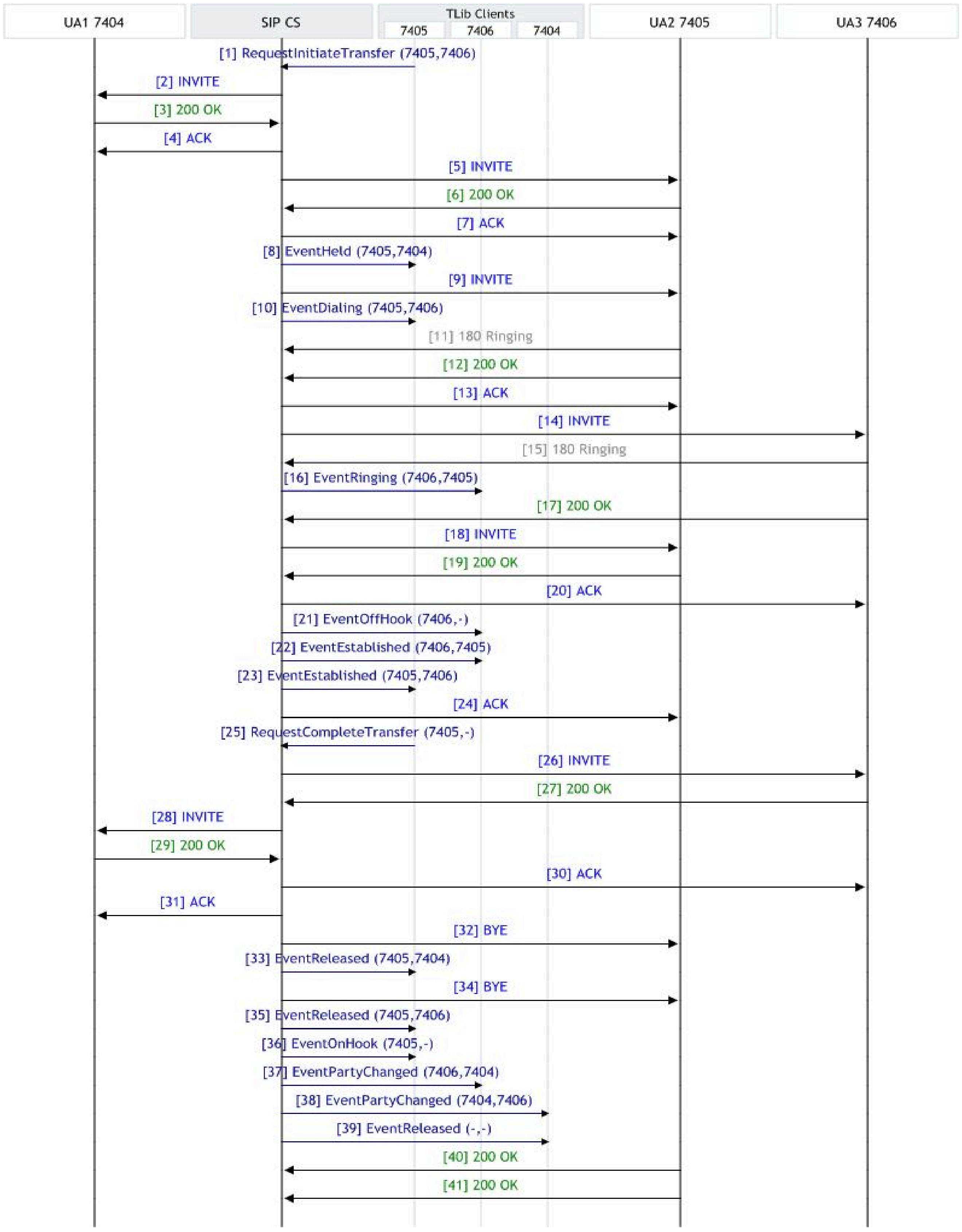
T-Lib-Initiated Consult Transfer re-INVITE

Description

SIP Server implements a TInitiateTransfer by invoking THoldCall and TMakeCall in a sequence.

The call scenario for TCompleteTransfer is shown here. It assumes a call from UA1 to UA2 is placed on hold and that a call from UA2 to UA3 is in the Established state.

Diagram


UA1 7404
SIP CS
TLib Clients
7405
7406
7404
UA2 7405
UA3 7406
[1] RequestInitiateTransfer (7405,7406)
[2] INVITE
[3] 200 OK
[4] ACK
[5] INVITE
[6] 200 OK
[7] ACK
[8] EventHeld (7405,7404)
[9] INVITE
[10] EventDialing (7405,7406)
[11] 180 Ringing
[12] 200 OK
[13] ACK
[14] INVITE
[15] 180 Ringing
[16] EventRinging (7406,7405)
[17] 200 OK
[18] INVITE
[19] 200 OK
[20] ACK
[21] EventOffHook (7406,-)
[22] EventEstablished (7406,7405)
[23] EventEstablished (7405,7406)
[24] ACK
[25] RequestCompleteTransfer (7405,-)
[26] INVITE
[27] 200 OK
[28] INVITE
[29] 200 OK
[30] ACK
[31] ACK
[32] BYE
[33] EventReleased (7405,7404)
[34] BYE
[35] EventReleased (7405,7406)
[36] EventOnHook (7405,-)
[37] EventPartyChanged (7406,7404)
[38] EventPartyChanged (7404,7406)
[39] EventReleased (-,-)
[40] 200 OK
[41] 200 OK

T-Lib-Initiated 3-Way Conf Central Mixing

Description

T-Lib–controlled three-way conferences are handled through an external multimedia conferencing unit (MCU) device that performs the audio mixing.

The call scenario begins when a call is established between UA1 and UA2. UA2 places UA1 on hold and dials a consult call to UA3. When UA2 and UA3 are ready for a conference, SIP Server re-Invites everybody to the mixing device. A DN (that is, the user part of a SIP URI) identifies a conference — all participants will use the same number to identify the conference to the mixing device.

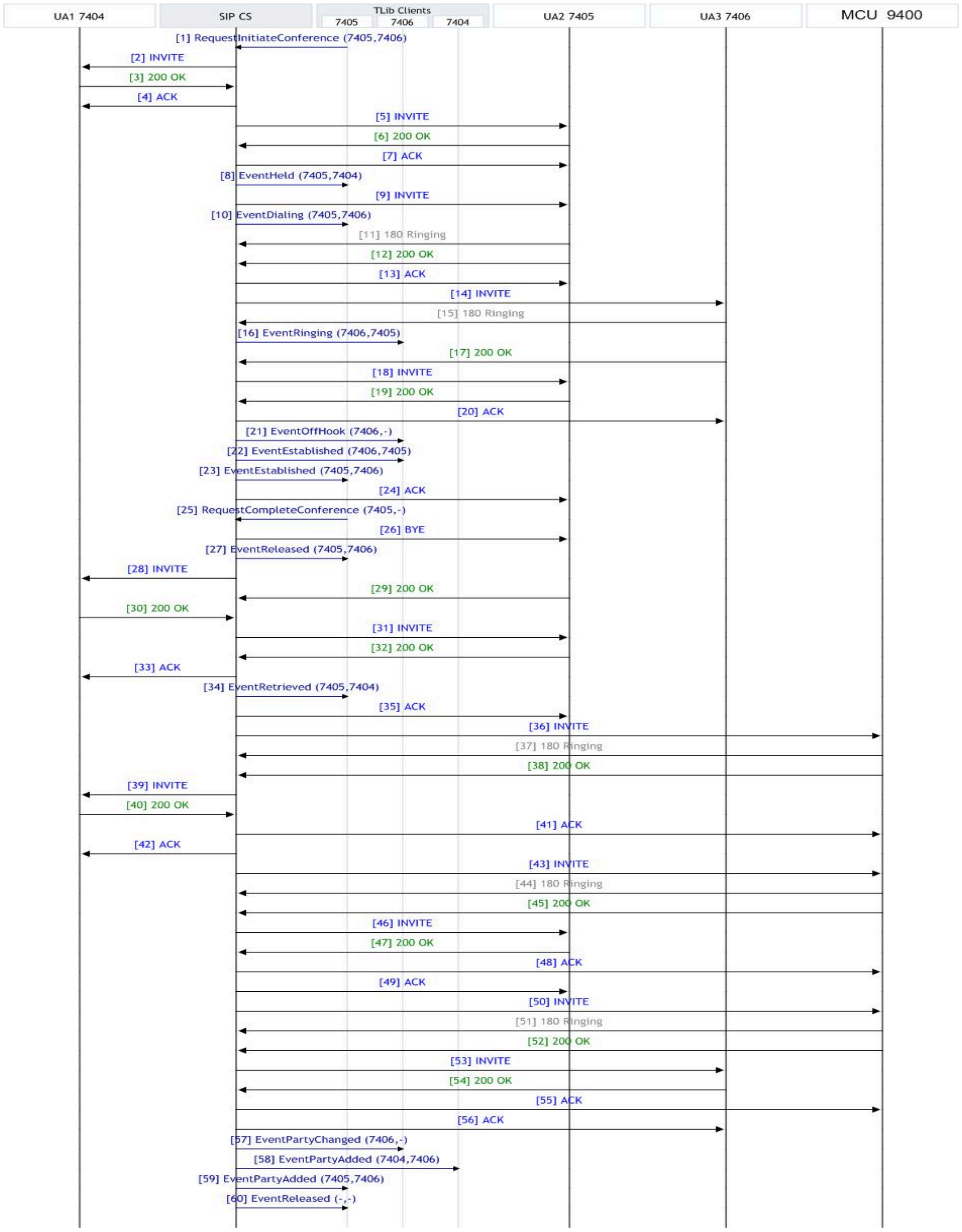
The diagram begins with a call between UA1 and UA2 in a Held state and a call from UA2 to UA3 in an Established state. Agent UA2 invokes the T-Lib request TCompleteConference.

Here is how this process works:

1. To allocate MCU resources, SIP Server starts by sending the MCU one INVITE request for each participant request without an SDP.
2. SIP Server sends INVITEs without the SDP offering to get the list of codecs from the MCU.
3. SIP Server does not modify any of the existing dialogs until it receives an OK for all INVITEs sent to the MCU. If any of the INVITEs fail, SIP Server rolls back to the initial state and all SIP dialogs remain intact.
4. Once SIP Server receives all the OKs, it sends re-INVITE dialogs to the three participants, using the SDPs it received from the MCU.
5. SIP Server uses the answer SDPs from the participants in the ACKs it sends to the MCU.
6. Before the conference is completed, UA2 will have had two dialogs — one for the primary call and one for the consult call. SIP Server re-INVITEs the primary call to the MCU; it sends BYE to terminate the consult call.

Implementation Note: The scenario just described is the most efficient way to establish a three-way conference using the T-Lib API. However, some MCU devices do not support INVITE without an SDP. To accommodate these devices, SIP Server sends INVITEs to MCU with valid non-hold SDPs (the actual SDP received from the corresponding devices). After that, it re-INVITEs the devices.

Diagram

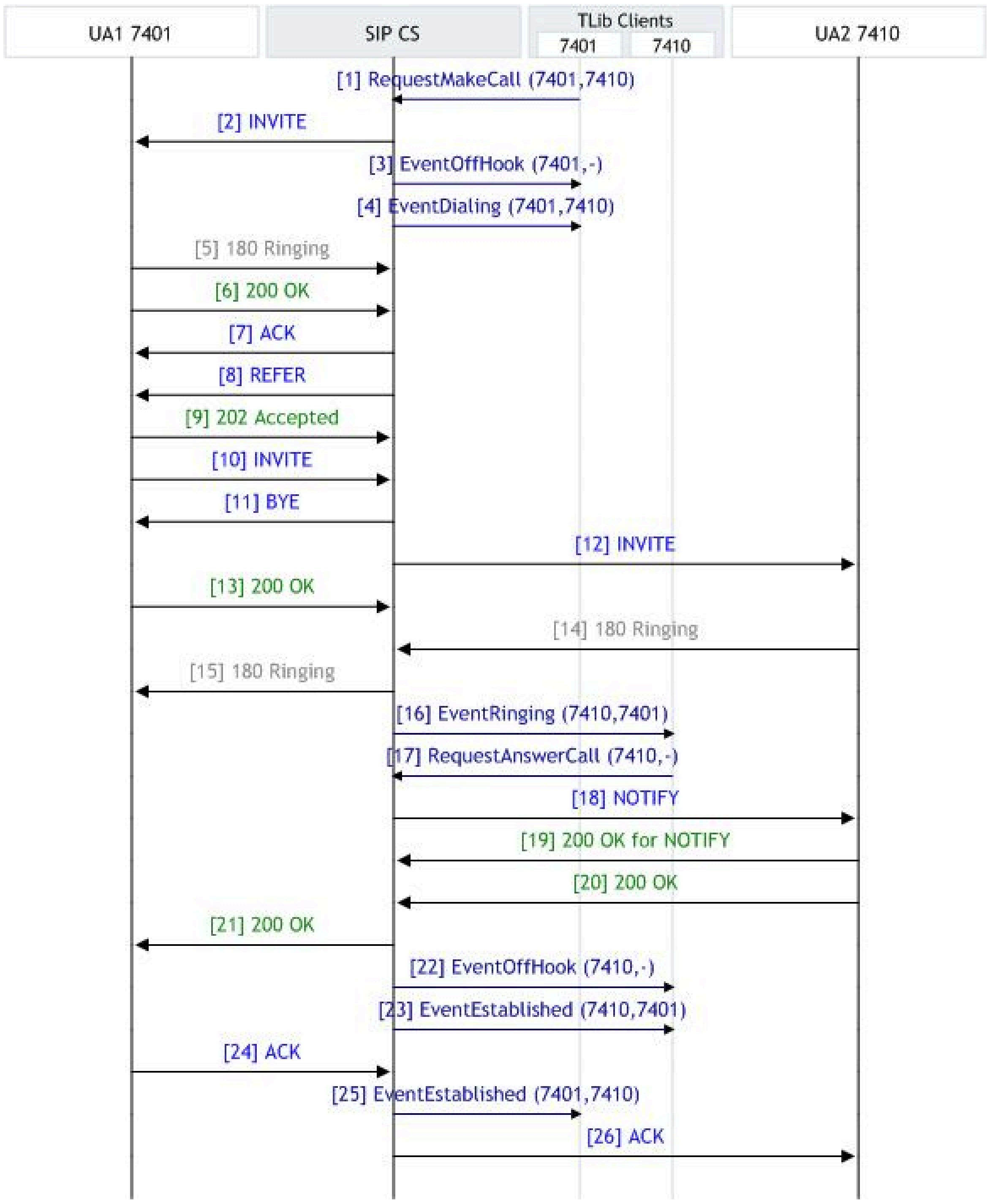

UA1 7404
SIP CS
TLib Clients
7405
7406
7404
UA2 7405
UA3 7406
MCU 9400
[1] RequestInitiateConference (7405,7406)
[2] INVITE
[3] 200 OK
[4] ACK
[5] INVITE
[6] 200 OK
[7] ACK
[8] EventHeld (7405,7404)
[9] INVITE
[10] EventDialing (7405,7406)
[11] 180 Ringing
[12] 200 OK
[13] ACK
[14] INVITE
[15] 180 Ringing
[16] EventRinging (7406,7405)
[17] 200 OK
[18] INVITE
[19] 200 OK
[20] ACK
[21] EventOffHook (7406,-)
[22] EventEstablished (7406,7405)
[23] EventEstablished (7405,7406)
[24] ACK
[25] RequestCompleteConference (7405,-)
[26] BYE
[27] EventReleased (7405,7406)
[28] INVITE
[29] 200 OK
[30] 200 OK
[31] INVITE
[32] 200 OK
[33] ACK
[34] EventRetrieved (7405,7404)
[35] ACK
[36] INVITE
[37] 180 Ringing
[38] 200 OK
[39] INVITE
[40] 200 OK
[41] ACK
[42] ACK
[43] INVITE
[44] 180 Ringing
[45] 200 OK
[46] INVITE
[47] 200 OK
[48] ACK
[49] ACK
[50] INVITE
[51] 180 Ringing
[52] 200 OK
[53] INVITE
[54] 200 OK
[55] ACK
[56] ACK
[57] EventPartyChanged (7406,-)
[58] EventPartyAdded (7404,7406)
[59] EventPartyAdded (7405,7406)
[60] EventReleased (-,-)

Answer Call Functionality

Description

If an endpoint (SIP phone) supports remote CTI control by providing the SIP extensions event package known as "BroadSoft" SIP extensions, SIP Server allows SIP Server clients to answer the call by means of the TAnswerCall request. When responding to the TAnswerCall request, SIP Server will send an event with the header "Event" and a header value of "talk" to the SIP phone, in a ringing state of NOTIFY. This event instructs the SIP phone to answer the call without human interference. Note that SIP phones that support such an event package should send a 180 Ringing message header of "Allow-Events:", which should contain "talk" inside its value.

Diagram


UA1 7401
SIP CS
TLib Clients
7401
7410
UA2 7410
[1] RequestMakeCall (7401,7410)
[2] INVITE
[3] EventOffHook (7401,-)
[4] EventDialing (7401,7410)
[5] 180 Ringing
[6] 200 OK
[7] ACK
[8] REFER
[9] 202 Accepted
[10] INVITE
[11] BYE
[12] INVITE
[13] 200 OK
[14] 180 Ringing
[15] 180 Ringing
[16] EventRinging (7410,7401)
[17] RequestAnswerCall (7410,-)
[18] NOTIFY
[19] 200 OK for NOTIFY
[20] 200 OK
[21] 200 OK
[22] EventOffHook (7410,-)
[23] EventEstablished (7410,7401)
[24] ACK
[25] EventEstablished (7401,7410)
[26] ACK