

BLISS
Internet-Draft
Intended status: Informational
Expires: September 5, 2007

M. Soroushnejad, Ed.
V. Venkataramanan
Sylantro Systems Corp
P. Pepper
Citel Technologies
A. Kumar
Yahoo Inc.
A. Johnston, Ed.
Avaya
March 4, 2007

**Implementing Multiple Line Appearances using the Session Initiation
Protocol (SIP)
draft-anil-sipping-bla-04**

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with [Section 6 of BCP 79](#).

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."

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/ietf/1id-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on September 5, 2007.

Copyright Notice

Copyright (C) The IETF Trust (2007).

Abstract

This document describes the implementation of a group telephony

feature commonly known as Bridged Line Appearance (BLA), Multiple Line Appearance (MLA), or Shared Call/Line Appearance (SCA). When implemented using the Session Initiation Protocol (SIP), it is referred to as Multiple Appearances. This feature is commonly offered in the IP Centrex services and IP-PBX offerings and is likely to be implemented on SIP IP telephones and SIP feature servers used in a business environment. This specification defines a new SIP dialog event package tag "appearance" and a method for a group of SIP user agents to coordinate the identification of appearances between them using SIP dialog package subscriptions.

Table of Contents

1.	Introduction	3
2.	Conventions used in this document	3
3.	Feature Description	3
3.1.	Usage Scenarios	5
4.	Overview of Operation	5
4.1.	Appearance Implementation	8
5.	Normative Description	10
5.1.	Multiple Appearance User Agents	10
5.2.	Appearance Agent	12
6.	Example Message Flows	13
6.1.	Registration and Subscription Flows	13
6.1.1.	Alice Registration and Subscription Flow	13
6.1.2.	Registration and Subscription Flow	17
6.2.	Call Originated within the Appearance Group	19
6.3.	Call Offered to an Appearance Group	30
6.4.	Use of PUBLISH	36
6.5.	Bridging on an Appearance	38
6.6.	State and Error Recovery Examples	40
6.6.1.	Line Seize (Refresh Subscription)	40
6.6.2.	Line Seize (Notifier Failure)	41
6.6.3.	Line Seize (Race Condition)	42
7.	IANA Considerations	42
8.	Security Considerations	43
9.	Acknowledgements	43
10.	Normative References	43
	Authors' Addresses	44
	Intellectual Property and Copyright Statements	46

1. Introduction

The feature and functionality requirements for SIP user agents supporting business telephony applications differ vastly from basic SIP user agents, both in terms of services and end user experience. In addition to basic SIP support, many of the services in a business environment require the support for SIP extensions such as REFER [3], SUBSCRIBE/NOTIFY primitives [4], the SIP Replaces header field [5], etc. Many of the popular business services have been documented in the SIPPING service-examples draft [6].

This specification details a method for implementing a group telephony feature known in telephony as Bridged Line Appearance (BLA) or Multiple Line Appearance (MLA), one of the more popular advanced features expected of SIP IP telephony devices in a business environment. Another common name for the this feature is Shared Call/Line Appearance (SCA).

This specification uses standard SIP [RFC 3261](#) [2] in conjunction with [RFC 3265](#) [4] for exchanging status among user agents, and the SIP dialog state event package [7] to exchange dialog state information to achieve the same. This specification also extends [RFC 4235](#) to add a new dialog package parameter "appearance" which is used to coordinate the appearance number of a given Address of Record (AOR). A parameter for the Alert-Info header field is also defined.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC-2119](#) [] and indicate requirement levels for compliant mechanisms.

3. Feature Description

The basic functionality of the multiple appearance feature can be summarized as follows:

1. Incoming calls to the AOR are offered to a group of UAs and can be answered by any of them. The group is referred to as BLA or MA group. The method by which the group is defined (i.e., created, members added/deleted, etc.) is out of the scope of this document.
2. Each UA in the group knows the call status of the others in the group and typically renders this information to the user.

3. Each call or session (incoming or outgoing) is assigned a unique "appearance" number from a managed pool administered for the AOR group. This number is used by the user interface for display of call information (e.g. the order of the lamp/button on the device) and out-of-band indications (e.g. "Sales pickup line 2, please"). Once the dialog has terminated, the appearance number is released back into the pool and can be reassigned to another incoming or outgoing session.

4. Calls can be joined (also called bridged or conferenced together) or can be placed on hold and picked up (taken) by another UA in the group.

#1 is commonly done today in SIP by having each UA register against the group AOR. Incoming requests are forked by the proxy server to all UAs.

#2 can be done using the SIP dialog package. To avoid a full mesh of dialog package subscriptions, this specification uses a State Agent.

#3 requires an extension to SIP, defined in this specification.

#4 can be done using the Replaces and Join header fields constructed using information obtained from the dialog package.

In traditional telephony, the line is physical. A common scenario is for a number of business telephones to share a single or a small number of lines. The appearance number relates to the user interface for the telephone - typically each appearance has a visual display (lamp that can change color or blink) and a button (used to select the appearance). In SIP terms, the line is virtual. However, the concept of appearance is still relevant to SIP due to the user interface considerations. It is important to keep the appearance number construct because:

1. Human users are used to the concept and will expect it in replacement systems (e.g. an overhead page announcement says "Joe pickup line 3")
2. It is a useful structure for user interface representation
3. There are cases where for bandwidth or gateway limitations, it is useful to limit the number of concurrent sessions.

In this document, we will use the term "appearance" as a stand-in for "line appearance". Note that this does not mean that a conventional telephony user interface (lamps and buttons) must be used - implementations may use another metaphor as long as the appearance

number is readily apparent to the user.

3.1. Usage Scenarios

The following examples are common applications of the Multiple Appearances feature and are mentioned here as informative use cases:

1. Executive/Assistant arrangement: The executive may appear as an appearance on the assistant SIP telephony device. Assistant may answer incoming calls to executive and then place the call on hold for the executive to pick it up. The assistant can always see the call state of the executive.
2. BLA Call Group: Users with similar business needs or tasks can be assigned to specific groups and share the line appearances of each other on each others SIP telephony devices. For example, an IT department staff of five might answer a help line which has three appearances on each phone in the IT work area. A call answered on one phone can be put on hold and picked up on another phone. A shout or an IM to another staff member can result in them taking over a call on a particular appearance.
3. Virtual BLA: Allows a AOR, not assigned as a primary appearance to any user, to be set up as a BLA on a given set of user devices.

All the example usages above can be supported by the Multiple Appearances feature described in this document, however the details of setup and usage of the above examples are not relevant to understanding of the BLA mechanism itself.

4. Overview of Operation

This section provides an overview of the components and operation of the service. It is non-normative in nature. The requirements for the Multiple Appearances feature are documented in [\[13\]](#).

The Multiple Line Appearance service uses the following components to enable this feature:

- SIP Registrar/Location Service
- SIP Forking Proxy
- Appearance Agent defined in this specification
- SIP User Agents that support this feature

The Appearance Agent implements a State Agent (SA) for the dialog state for the AOR of the group and manages the appearance number shared resource.

Figure 1 illustrates the SIP components involved in supporting the Multiple Appearance feature as described in this document.

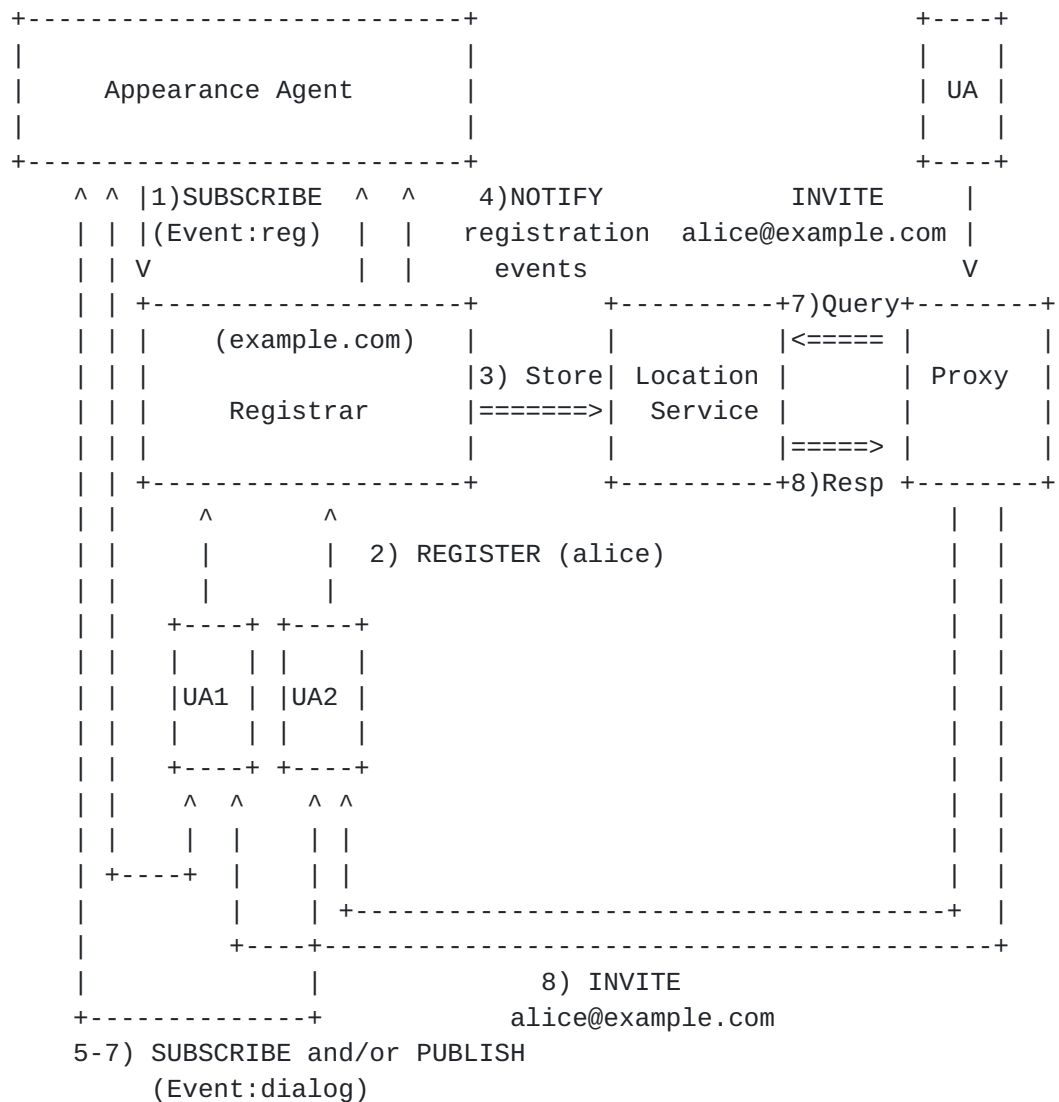


Figure 1.

1. The Appearance Agent SUBSCRIBES to the registration event package as outlined in [8] for contacts registered to the group AoR. Thus, it has knowledge of all User Agents registered against the AOR at any point of time.

2. User Agents (UA1 and UA2 in Figure 1) belong to the appearance

group and REGISTER against the same AOR (e.g., `alice@example.com`). The SIP third-party registration mechanism (i.e., when From: is not equal to To: in the REGISTER message) can be used to allow the members of the group to have different authentication credentials.

3. Each registration is stored in the Location Service.
4. The registrar notifies the Appearance Agent of successful registration at each UA.
5. The Appearance Agent SUBSCRIBES to User Agents for the state of all dialogs associated with the UA1 and UA2, as defined in [7]. Alternatively, if configured with the URI of the Appearance Agent, UA1 and UA2 can PUBLISH [14] their dialog state directly.
6. The User Agents SUBSCRIBE to the Appearance Agent for the state of all dialogs as defined in [7]. The Request-URI of the SUBSCRIBE could be either the AOR of the group or the Contact URI information it received in the incoming subscription from the Appearance Agent.
7. The User Agents act as the notifier for the dialog state events as defined in [7]. They send a NOTIFY every time their dialog state changes (i.e. receive an INVITE, enter alerting state, answer a call, terminate a call, generate an INVITE, etc.)
8. Forking Proxy forks an incoming INVITE for the AOR address to the registered user agents.

The User Agents in the group could SUBSCRIBE to each other and NOTIFY dialog state events, but in a large group the User Agents have to manage a larger number of SUBSCRIPTIONS and NOTIFICATIONS. The State Agent in the Appearance Agent helps in managing large groups better. Further, the State Agent can filter dialog state events and NOTIFY User Agents of the dialog state events which are required for the application or feature. The State Agent can also SUBSCRIBE to dialog state events with filters to reduce the number of NOTIFY messages exchanged between the State Agent and the user agents in the group. This allows a group of N UAs to each only establish a single dialog state subscription to learn the dialog state of all other group members. This results in 2N total subscriptions for the entire group. A full mesh of subscriptions without a state agent would result in $N(N-1)$ total subscriptions.

Just as conferencing systems commonly have a single point of control, known as a focus, a Multiple Appearance group has a single point of control of the appearance shared resource. This is known as the Appearance Agent for a group. While an Appearance Agent can be part of a centralized server, it could also be co-resident in a member

User Agent who has taken on this functionality for a group. The Appearance Agent learns the group state either by subscribing to the dialog state of each member UA individually or by dialog state publications from members.

The Appearance Agent can select the appearance number for an incoming call. The appearance number is a non-negative integer: 0,1,2, etc. An Appearance agent can use the registration event package to learn how many UAs are part of the group. The Appearance Agent sends a NOTIFY of dialog state events to all the User Agents.

4.1. Appearance Implementation

This section discusses and compares two methods of implementing the appearance function. One uses a URI parameter while the other uses a SIP dialog package extension parameter.

Some implementations of this feature utilize a URI parameter such as "line=3" on the Contact URI. While this does work, it is sub-optimal for the following reasons:

1. Each "line" on a UA needs to REGISTER individually against the AOR. For a system of many multi-line phones, this greatly multiplies the number of registrations and refreshes. For N phones each with n appearances, this means nN registrations.
2. Since each "line" has a unique Contact, a separate dialog package subscription would be needed for each "line". This would result in 2nN subscriptions.
3. The number of appearances is controlled by the UA. Using other approaches, it is possible to have the Appearance Agent dynamically manage the number of appearances.
4. The "line" number conveyed in the Contact URI would be always shared with the other UA in the dialog, possibly leaking private information ("So who do you have on line 1?").
5. Incoming INVITEs constructed using Contact URIs could be rejected due to having incorrect "line" number settings.

Instead of the URI parameter approach, this specification defines an extension parameter "appearance" to the SIP dialog package. The value of this parameter is a non-negative integer: 0,1,2, etc. Compared to the URI parameter approach:

1. Only one registration per UA per AOR is required, as per normal SIP.

2. Only a single dialog package subscription per AOR per UA is needed, resulting in 2N total subscriptions.
3. The number of appearances is centrally managed and controlled by the Appearance Agent.
4. The appearance number is never leaked to the other UA in a dialog.
5. Only the Appearance Agent can select an appearance number for incoming requests, avoiding call failures.

When a UA indicates, via a dialog-info NOTIFY command, a state change on a dialog, the appearance parameter is placed in the parameters of the local target of the dialog-info. For example:

```
<?xml version="1.0"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
    version="6"
    state="partial"
    entity="sip:alice@example.com">
  <dialog id="id3d4f9c83" direction="initiator">
    <state>trying</state>
    <local>
      <target uri="sip:bob@example.com">
        <param pname="appearance" pvalue="0"/>
      </target>
    </local>
  </dialog>
</dialog-info>
...
```

Before the user can be alerted on an incoming request, a UA needs to know the appearance number for an incoming request. For this reason, an extension parameter is defined for the Alert-Info header field.

For example:

Alert-Info: <file://ring.pcm>;alert=normal;appearance=0

This Alert-Info header would indicate to place the call on the first line appearance instance.

The determination as to what value to use in the appearance parameter can be done at the proxy that forks the incoming request to all the registered UA's. There are a variety of ways the proxy can use to determine what value it should use to populate this parameter. For

example, the proxy could fetch this information by initiating a SUBSCRIBE request with Expires: 0 for the AOR to fetch the list of lines that are in use. Alternatively, it could act like a UA that is a part of the APPEARANCE group and SUBSCRIBE to the State-Agent like any other UA. This would ensure that the active dialog information is available without having to poll on a need basis. It could keep track of the list of active calls for the APPEARANCE AOR based on how many unique INVITE requests it has forked to or received from the APPEARANCE AOR. The actual mechanism is left to the implementation.

The Appearance Agent needs to know about all incoming requests to the AOR in order to select the appearance number. One way in which this could be done is for the Appearance Agent to register against the AOR with a higher q value. This will result in the INVITE being sent to the Appearance Agent first, then being offered to the UAs in the group. The actual mechanism used is left to the implementation.

It should be noted that the appearance parameter is relative to the AOR. Thus a UA can support multiple AOR's with each AOR capable of supporting appearance's sequentially numbered from 0 through n-1 where n is the number of lines the UA supports for each of the AOR's.

5. Normative Description

This section normatively describes the service.

5.1. Multiple Appearance User Agents

User Agents that support the Multiple Appearance feature MUST support the dialog state package as outlined in [\[7\]](#) to NOTIFY other User Agents of the dialog-state.

User Agents MUST support Replaces [\[5\]](#) and MAY support Join [\[11\]](#).

They MUST support the "appearance" extensions to the dialog package defined in this specification. They SHOULD understand and support the (ma) event parameter outlined in this draft. [Section 7](#) of this draft outlines how this parameter is to be interpreted by the UA.

When generating dialog package notifications, UAs MUST include dialog identification information including call-id, to-tag and from-tag. When calls are placed on hold, the UAs MAY include local session description in the dialog package notification. The element should indicate that the call that is the subject of the dialog is being held (use of a=inactive or a=sendonly is RECOMMENDED [\[9\]](#)). When calls are placed on hold, the "+sip.rendering=no" [\[RFC 3840\]](#) feature tag MUST be used.

The UA MUST send dialog-info in state "trying" with the appropriate appearance parameter present in the local target when it is attempting to 'seize' a line appearance. It MUST NOT send the INVITE until a 200 OK response to the NOTIFY is received from the Appearance Agent.

Note: Sending NOTIFY dialog state of (trying) before an INVITE is a departure from [7], but this MUST be implemented to resolve glare issues.

In case a UA receives a 500 final response for the NOTIFY, it MUST inspect the Retry-After interval. If one is present, the UA must wait for this interval before it retries sending the NOTIFY. Further it should continue to delay sending out the INVITE. Should the UA receive a NOTIFY of (trying) from the Appearance Agent before the expiry of this interval, it MUST honor the same by providing appropriate end user indication; cancel any timers it has started for retrying the NOTIFY request; and abandon reinitiating of the NOTIFY request. The UA MUST then abandon sending out INVITE to the address in such a scenario. It should be noted that this 500 response does not amount to a NOTIFY failure. Specifically, this response MUST not result in the UA terminating Subscriptions of the Appearance Agent.

This is needed to resolve conflicts in the use of a particular appearance.

Further, the dialog state definition [7] has no restrictions on the number of dialogs that MAY be conveyed in a single SIP NOTIFY message. However, since the NOTIFY for going off-hook can be rejected by the Appearance Agent, such a NOTIFY MUST be presented to the Appearance Agent as a single dialog payload in the NOTIFY.

The dialog state package defined in [7] defines the set of messages that MAY be provided by a UA to publish state information of dialogs. For satisfactory operation of this feature, the flows require that the UA SUBSCRIBE to the dialog-event package on receipt of a SUBSCRIBE request. It MUST use the contact in this SUBSCRIBE for making 'line reservations' when placing calls from the AOR. Also, as seen in the above example message flows, not all of these messages need to be published by a UA to effectively participate in a BLA group. In order to indicate this preference, this draft proposes a new Event parameter for the dialog package called (ma). User Agents that understand this parameter will provide dialog state information as detailed in this draft.

A critical aspect for reliable operation of this feature is the ability for all user agents in the BLA group to recover from network failures caused at a single UA. For example, one of the user agents

in the BLA group may have answered an incoming call, notified the dialog state to other members and then experienced a network outage. The calling UA could have detected the same (using RTP or some other means) and could have hung up. However, none of the other user agents in the BLA group would get notification of this change in dialog state and their BLA appearances could stay out of sync for a long time; depending on when the network is restored, or when the Appearance Agent attempts to refresh its dialog-state subscription with the failed UA. To recover from such a failure, the Appearance Agent **MUST** **SUBSCRIBE** with a shorter expiration (expiration interval not smaller than 300 seconds is **RECOMMENDED**) following the notification of a "confirmed" dialog or when a Event Agent honors a "trying" for call origination, with the user agents that notified it of this information.

Alternatively, while a user agent is acquiring, or holds a line seize, the dialog subscriptions to it act as BLA status indicators to the subscriber. If subscription refresh requests to the user agent are not honored, then it must be determined that the user agent's capacity to maintain line seize has been lost. For this reason, during the period a user agent is acquiring or holds a line seize, the remaining expiry period of subscriptions to it **MAY** be reduced by the user agent to minimize the outage problem (expiration interval not smaller than 300 seconds is **RECOMMENDED**). This can be accomplished by setting the Expires parameter in the Subscription-State header in the NOTIFY message sent out by the UA.

5.2. Appearance Agent

An Appearance Agent defined in this specification **MUST** implement a dialog package state agent for the UAs registered against the AOR.

The Appearance Agent **MUST** support the appearance dialog package extensions defined in this specification.

Even in trivial deployments of two multiple appearance enabled user agents, race conditions can exist when both user agents attempt to utilize an appearance simultaneously (i.e. when the NOTIFY messages, that each user agent sends to the other, are active within the network during an overlapping time period). The SIP-specific event notification framework, described in [4], defines the use of state agents. State agents act on behalf of resources to publish state. The Appearance Agent can use any policy deemed necessary to resolve races and ensure no two user agents obtain the same line seize during overlapping periods.

Appearance Agents are responsible for resolving conflicts in the appearance resource. If a UA requests the use of an appearance

number that is in use by another UA, the Appearance Agent will send a 500 response and resend a dialog state notification to the UA to allow them to synchronize with the proper state. An example is shown in [Section 7.3](#).

6. Example Message Flows

The next section shows call flows. The flows and descriptions are non-normative.

6.1. Registration and Subscription Flows

Bob has bridged line appearance for Alice. Bob REGISTERS for Alice's AOR using contact sip:alice@ua2.example.com. Furthermore, Bob REGISTERS his primary address with contact sip: bob@ua2.example.com. Alice REGISTERS with contact sip:alice@ua1.example.com.

The Appearance Agent subscribes to dialog states of the BLA AOR (i.e., sip:alice@example.com) at the User Agents for Alice and Bob. Message exchange between the Registrar, Appearance Agent, Alice, and Bob are shown below. The call flow examples below do not show challenging of subscriptions or notifications. It should be noted that for security purposes, all subscriptions MUST be authorized before the same is accepted.

6.1.1. Alice Registration and Subscription Flow

Registrar	Appearance Agent	Alice
<----- REGISTER F1<		
>F2 200 OK ----->		
	>F3 SUBSCRIBE ----->	
	<----- 200 OK F4<	
	<----- NOTIFY F5<	
	>F6 200 OK ----->	
	<----- SUBSCRIBE F7<	
	>F8 202 Accepted -->	


```
|>F9 NOTIFY ----->|
|                         |
|<----- OK 200 F10<|
|                         |
```

F1-F2: Alice registers AOR with contact: <sip:alice@ua1.example.com>

F1 Alice ----> Registrar

```
REGISTER sip:registrar.example.com SIP/2.0
Via: SIP/2.0/UDP ua1.example.com;branch=z9hG4bK527b54da8ACC7B09
From: <sip:alice@example.com>;tag=CDF9A668-909E2BDD
To: <sip:alice@example.com>
CSeq: 2 REGISTER
Call-ID: d3281184-518783de-cc23d6bb@ua1.example.com
Contact: <sip:alice@ua1.example.com>
User-Agent: ABC-UA/1.2.3
Max-Forwards: 70
Expires: 3600
Content-Length: 0
```

F2 Registrar ----> Alice

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP ua1.example.com;branch=z9hG4bKfbf176ef7F1D5FA2
CSeq: 2 REGISTER
Call-ID: d3281184-518783de-cc23d6bb@ua1.example.com
From: <sip:alice@example.com>;tag=CDF9A668-909E2BDD
To: <sip:alice@example.com>;tag=1664573879820199
Contact: <sip:alice@ua1.example.com>
Expires: 3600
Content-Length: 0
```

F3 to F6: Once Alice registers, Appearance Agent subscribes to the events at the contact registered for Alice and Alice notifies the Appearance Agent of its status.

F3 Appearance Agent ----> Alice

```
SUBSCRIBE sip:alice@ua1.example.com SIP/2.0
From: <sip:alice@example.com>;tag=110286377866002
To: <sip:alice@example.com>
Call-ID: 284-425690188@example.com
CSeq: 2 SUBSCRIBE
```


Via: SIP/2.0/UDP stateagent.example.com;branch=z9hG4bK1979345546866532
Event: dialog;ma
Expires: 3700
Contact: <sip:sa@stateagent.example.com>
Content-Length: 0

F4 Alice ----> Appearance Agent

SIP/2.0 200 OK
Via: SIP/2.0/UDP stateagent.example.com;branch=z9hG4bK1979345546866532
From: <sip:alice@example.com>;tag=110286377866002
To: <sip:alice@example.com>;tag=717A8C26-BA734CE3
CSeq: 2 SUBSCRIBE
Call-ID: 284-425690188@example.com
Contact: <sip:alice@ua1.example.com>
Event: dialog;ma
User-Agent: ABC-UA/1.2.3
Expires: 3700
Content-Length: 0

F5 Alice ----> Appearance Agent

NOTIFY sip:sa@stateagent.example.com SIP/2.0
Via: SIP/2.0/UDP ua1.example.com;branch=z9hG4bKa2269cc565A30870
From: <sip:alice@example.com>;tag=717A8C26-BA734CE3
To: <sip:alice@example.com>;tag=110286377866002
CSeq: 1 NOTIFY
Call-ID: 284-425690188@example.com
Contact: <sip:alice@ua1.example.com>
Event: dialog;ma
User-Agent: ABC-UA/1.2.3
Subscription-State: active;expires=3698
Max-Forwards: 70
Content-Type: application/dialog-info+xml
Content-Length: 164

```
<?xml version="1.0"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
              version="0"
              state="full"
              entity="sip:alice@example.com">
</dialog-info>
```

F6 Appearance Agent ----> Alice

SIP/2.0 200 OK
Via: SIP/2.0/UDP ua1.example.com;branch=z9hG4bKa2269cc565A30870
CSeq: 1 NOTIFY
Call-ID: 284-425690188@example.com
From: <sip:alice@example.com>;tag=717A8C26-BA734CE3
To: <sip:alice@example.com>;tag=110286377866002
Allow-Events: dialog;ma
Contact: <sip:sa@stateagent.example.com>
Content-Length: 0

F7 to F10: Alice also subscribes to the events associated with the Appearance AOR. Appearance Agent also notifies Alice of the status.

F7 Alice ----> Appearance Agent

SUBSCRIBE sip:sa@stateagent.example.com SIP/2.0
Via: SIP/2.0/UDP ua1.example.com;branch=z9hG4bKf10fac97E7A76D6A
From: <sip:alice@example.com>;tag=925A3CAD-CEBB276E
To: <sip:sa@stateagent.example.com>
CSeq: 1 SUBSCRIBE
Call-ID: ef4704d9-bb68aa0b-474c9d94@ua1.example.com
Contact: <sip:alice@ua1.example.com>
Event: dialog;ma
User-Agent: ABC-UA/1.2.3
Accept: application/dialog-info+xml
Max-Forwards: 70
Expires: 3700
Content-Length: 0

F8 Appearance Agent ----> Alice

SIP/2.0 202 Accepted
Via: SIP/2.0/UDP ua1.example.com;branch=z9hG4bKf10fac97E7A76D6A
CSeq: 1 SUBSCRIBE
Call-ID: ef4704d9-bb68aa0b-474c9d94@ua1.example.com
From: <sip:alice@example.com>;tag=925A3CAD-CEBB276E
To: <sip:sa@stateagent.example.com>;tag=1636248422222257
Allow-Events: dialog;ma
Expires: 3700
Contact: <sip:sa@stateagent.example.com>
Content-Length: 0

F9 Appearance Agent ----> Alice

NOTIFY sip:alice@ua1.example.com SIP/2.0


```

From: <sip:sa@stateagent.example.com>;tag=1636248422222257
To: <sip:alice@example.com>;tag=925A3CAD-CEBB276E
Call-ID: ef4704d9-bb68aa0b-474c9d94@ua1.example.com
CSeq: 2 NOTIFY
Via: SIP/2.0/UDP stateagent.example.com;branch=z9hG4bK1846984327225734
Max-Forwards: 70
Content-Type: application/dialog-info+xml
Event: dialog;ma
Subscription-State: active
Contact: <sip:sa@stateagent.example.com>
Content-Length: 162

```

```

<?xml version="1.0"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
              version="40"
              state="full"
              entity="sip:alice@example.com">
</dialog-info>

```

F10 Alice ----> Appearance Agent

```

SIP/2.0 200 OK
Via: SIP/2.0/UDP stateagent.example.com;branch=z9hG4bK1846984327225734
From: <sip:sa@stateagent.example.com>;tag=1636248422222257
To: <sip:alice@example.com>;tag=925A3CAD-CEBB276E
CSeq: 2 NOTIFY
Call-ID: ef4704d9-bb68aa0b-474c9d94@ua1.example.com
Contact: <sip:alice@ua1.example.com>
Event: dialog;ma
User-Agent: ABC-UA/1.2.3
Content-Length: 0

```

6.1.2. Registration and Subscription Flow

Registrar	Appearance Agent	Bob
<-----	REGISTER F1<	
>F2 200 OK ----->		
<-----	REGISTER F3<	
>F4 200 OK ----->		


```

|                                     |>F5 SUBSCRIBE ----->|
|                                     |                             |
|                                     |<----- 200 OK F6<|
|                                     |                             |
|                                     |<----- NOTIFY F7<|
|                                     |                             |
|                                     |>F8 200 OK ----->|
|                                     |                             |
|                                     |<----- SUBSCRIBE F9<|
|                                     |                             |
|                                     |>F10 202 Accepted ->|
|                                     |                             |
|                                     |>F11 NOTIFY ----->|
|                                     |                             |
|                                     |<----- 200 OK F12<|
|                                     |                             |

```

F1 to F2: Bob registers his (private) AOR with contact sip:bob@ua2.example.com (i.e., first-party registration).

F1 Bob ----> Registrar

```

REGISTER sip:registrar.example.com SIP/2.0
Via: SIP/2.0/UDP ua2.example.com;branch=z9hG4bK41599ad63A2E521F
From: <sip:bob@example.com>;tag=9F80647C-94355FE3
To: <sip:bob@example.com>
CSeq: 2 REGISTER
Call-ID: e4a1e078-75ccf65a-1a4c1ee1@ua2.example.com
Contact: <sip:bob@ua2.example.com>
User-Agent: XYZ-UA/4.5.6
Max-Forwards: 70
Expires: 3600
Content-Length: 0

```

F2 Registrar ----> Bob

```

SIP/2.0 200 OK
Via: SIP/2.0/UDP ua2.example.com;branch=z9hG4bK1b1c1d25FB82B2DE
CSeq: 2 REGISTER
Call-ID: e4a1e078-75ccf65a-1a4c1ee1@ua2.example.com
From: <sip:bob@example.com>;tag=9F80647C-94355FE3
To: <sip:bob@example.com>;tag=468305689550907
Contact: <sip:bob@ua2.example.com>
Expires: 3600
Content-Length: 0

```


F3 to F4: Bob registers Alice AOR with his client using SIP third-party registration. Note that this is considered third-party since From is different from To in the REGISTER.

F3 Bob ----> Registrar

```
REGISTER sip:registrar.example.com SIP/2.0
Via: SIP/2.0/UDP ua2.example.com;branch=z9hG4bK5f3f2c694DF31062
From: <sip:bob@example.com>;tag=81B52F2F-7EA64EE6
To: <sip:alice@example.com>
CSeq: 1 REGISTER
Call-ID: 55d48e6b-a0598cad-9ab32f84@ua2.example.com
Contact: <sip:alice@ua2.example.com>
User-Agent: XYZ-UA/4.5.6
Max-Forwards: 70
Expires: 3600
Content-Length: 0
```

F4 Registrar ----> Bob

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP ua2.example.com;branch=z9hG4bKdf17f688391F7DF1
CSeq: 2 REGISTER
Call-ID: 55d48e6b-a0598cad-9ab32f84@ua2.example.com
From: <sip:bob@example.com>;tag=81B52F2F-7EA64EE6
To: <sip:alice@example.com>;tag=773736136499990
Contact: <sip:alice@ua2.example.com>
Expires: 3600
Content-Length: 0
```

F5 to F10: Once Bob registers with Alice AOR, Appearance Agent subscribes to the events at the contact registered for Alice and Bob notifies the Appearance Agent of its status. These messages are not shown as they are essentially identical to the previous call flow.

6.2. Call Originated within the Appearance Group

In this scenario, the UA just before allowing the user to place a call, sends out a dialog event NOTIFY with state (trying). Only after receiving the 200 OK does the UA proceed with the call and send the INVITE.

In the following call flow, Bob, as a member of the Alice BLA group, places an outgoing call to Carol using Alice line appearance. Bob then places Carol on hold. Alice subsequently picks up the held call

and has a established session with Carol. Finally, Carol hangs up. The details of the notifications amongst the user agents and the Appearance Agent in updating the status of the BLA group members are shown below. For brevity, details of some of the messages are not included in the message flows.

Carol	Proxy	Alice	Appearance Agent	Bob
			<----- NOTIFY F1<	
			>F2 200 OK ----->	
		<-- NOTIFY F3<		
		>F4 200 OK -->		
	<----- INVITE F5<			
<-- INVITE F6<				
>F7 180 Ring >				
	>F8 180 Ringing ----->			
>F9 200 OK -->				
	>F10 200 OK ----->			
	<----- ACK F11<			
<===== Both way RTP established =====>				
			<----- NOTIFY F12<	
			>F13 200 OK ----->	
		<- NOTIFY F14<		
		>F15 200 OK ->		
	<----- (hold) INVITE F16<			
<- INVITE F17<				
>F18 200 OK ->				
	>F19 200 OK ----->			
	<----- ACK F20<			
			<----- NOTIFY F21<	
			>F22 200 OK ----->	


```

|                                     |<- NOTIFY F23<|
|                                     |
|                                     |>F24 200 OK ->|
|                                     |
|<-- INVITE F25<|
|<- INVITE F26<|(w/ Replaces) |
| ( w/ Replaces)|
|>F27 200 OK ->|
|                                     |>F28 200 OK -->|
|                                     |
|<----- ACK F29<|
|                                     |
|<= Both way RTP established =>|
|                                     |
|>F30 BYE ---->|
|                                     |>F31 BYE ----->|
|                                     |
|                                     |<----- OK 200 F32<|
|<- 200 OK F33<|
|                                     |
|                                     |<----- NOTIFY F34<|
|                                     |
|                                     |>F35 200 OK ---->|
|                                     |
|                                     |<- NOTIFY F36<|
|                                     |
|                                     |>F37 200 OK ->|
|                                     |
|                                     |>F38 NOTIFY ->|
|                                     |
|                                     |<- 200 OK F39<|
|                                     |>F40 NOTIFY ---->|
|                                     |
|                                     |<----- 200 OK F41<|
|>F42 BYE ---->|
|                                     |>F43 BYE ---->|
|                                     |
|                                     |<-- 200 OK F44<|
|<--200 OK F45<|
|                                     |>F46 NOTIFY ->|
|                                     |
|                                     |<- 200 OK F47<|
|                                     |
|                                     |>F48 NOTIFY ---->|
|                                     |
|                                     |<----- OK 200 F49<|

```

F1 to F4: Bob uses the BLA appearance of Alice on his UA to place an outgoing call (e.g., he goes off-hook). Before sending the outgoing INVITE request, Bob notifies the same agent that Alice line appearance is in (trying) state. Appearance Agent notifies Alice of

the same event by forwarding the NOTIFY payload provided by Bob after appropriately changing the dialog id field in the XML payload to a unique value towards each of the entities it is forwarding to (Alice in this example).

F1 Bob ----> Appearance Agent

```
NOTIFY sip:sa@stateagent.example.com SIP/2.0
Via: SIP/2.0/UDP ua2.example.com;branch=z9hG4bK61314d6446383E79
From: <sip:bob@example.com>;tag=44150CC6-A7B7919D
To: <sip:alice@example.com>;tag=428765950880801
CSeq: 7 NOTIFY
Call-ID: 144-1289338424@example.com
Contact: <sip:alice@ua2.example.com>
Event: dialog;ma
User-Agent: XYZ-UA/4.5.6
Subscription-State: active;expires=3347
Max-Forwards: 70
Content-Type: application/dialog-info+xml
Content-Length: 365
```

```
<?xml version="1.0"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
  version="6"
  state="partial"
  entity="sip:alice@example.com">
  <dialog id="id3d4f9c83" direction="initiator">
    <state>trying</state>
    <local>
      <target uri="sip:bob@example.com">
        <param pname="appearance" pvalue="0" />
      </target>
    </local>
  </dialog>
</dialog-info>
```

F2 Appearance Agent ----> Bob

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP ua2.example.com;branch=z9hG4bK61314d6446383E79
CSeq: 7 NOTIFY
Call-ID: 144-1289338424@example.com
From: <sip:bob@example.com>;tag=44150CC6-A7B7919D
To: <sip:alice@example.com>;tag=428765950880801
Allow-Events: dialog;ma
Contact: <sip:sa@stateagent.example.com>
```


Content-Length: 0

F3 Appearance Agent ----> Alice

NOTIFY sip:alice@ua1.example.com SIP/2.0
From: <sip:alice@example.com>;tag=497585728578386
To: <sip:alice@example.com>;tag=633618CF-B9C2EDA4
Call-ID: a7d559db-d6d7dcad-311c9e3a@ua1.example.com
CSeq: 7 NOTIFY
Via: SIP/2.0/UDP stateagent.example.com;branch=z9hG4bK1711759878512309
Max-Forwards: 70
Content-Type: application/dialog-info+xml
Event: dialog;ma
Subscription-State: active
Contact: <sip:sa@stateagent.example.com>
Content-Length: 402

```
<?xml version="1.0"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
  version="27"
  state="partial"
  entity="sip:alice@example.com">
  <dialog id="fa02538339df3ce597f9e3e3699e28fc"
    direction="initiator">
    <state>trying</state>
    <local>
      <target uri="sip:bob@example.com">
        <param pname="appearance" pvalue="0"/>
      </target>
    </local>
  </dialog>
</dialog-info>
```

F4 Alice ----> Appearance Agent

SIP/2.0 200 OK
Via: SIP/2.0/UDP stateagent.example.com;branch=z9hG4bK1711759878512309
From: <sip:alice@example.com>;tag=497585728578386
To: <sip:alice@example.com>;tag=633618CF-B9C2EDA4
CSeq: 7 NOTIFY
Call-ID: a7d559db-d6d7dcad-311c9e3a@ua1.example.com
Contact: <sip:alice@ua1.example.com>
Event: dialog;ma
User-Agent: ABC-UA/1.2.3
Content-Length: 0

F5 to F11: Bob places a call to Carol by sending the INVITE request towards the Proxy. The INVITE (see F5 message below) includes a P-Preferred-Identity header [10] to designate the identity to be used as the calling party for this call (i.e., Alice instead of Bob).

F5 Bob ----> Proxy

```
INVITE sip:carol@example.com SIP/2.0
Via: SIP/2.0/UDP ua2.example.com;branch=z9hG4bK98c87c52123A08BF
From: <sip:bob@example.com>;tag=15A3DE7C-9283203B
To: <sip:carol@example.com>
CSeq: 1 INVITE
Call-ID: f3b3cbd0-a2c5775e-5df9f8d5@ua2.example.com
Contact: <sip:alice@ua2.example.com>
User-Agent: XYZ-UA/4.5.6
P-Preferred-Identity: <sip:alice@example.com>
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 223
```

```
v=0
o=- 1102980499 1102980499 IN IP4 ua2.example.com
s=IP SIP UA
c=IN IP4 ua2.example.com
t=0 0
a=sendrecv
m=audio 2236 RTP/AVP 0 8 101
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:101 telephone-event/8000
```

F12 to F15: Bob notifies the Appearance Agent of the status of the dialog (i.e., confirmed). Appearance Agent notifies Alice of the same.

F12 Bob ----> Appearance Agent

```
NOTIFY sip:sa@stateagent.example.com SIP/2.0
Via: SIP/2.0/UDP ua2.example.com;branch=z9hG4bKa39d3f69D4E20602
From: <sip:bob@example.com>;tag=44150CC6-A7B7919D
To: <sip:alice@example.com>;tag=428765950880801
CSeq: 9 NOTIFY
Call-ID: 144-1289338424@example.com
Contact: <sip:alice@ua2.example.com>
Event: dialog;ma
User-Agent: XYZ-UA/4.5.6
```


Subscription-State: active;expires=3342
Max-Forwards: 70
Content-Type: application/dialog-info+xml
Content-Length: 675

```
<?xml version="1.0"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
  version="8"
  state="partial"
  entity="sip:alice@example.com">
  <dialog id="id3d4f9c83"
    call-id="f3b3cbd0-a2c5775e-5df9f8d5@ua2.example.com"
    local-tag="15A3DE7C-9283203B"
    remote-tag="65a98f7c-1dd2-11b2-88c6-b03162323164+65a98f7c"
    direction="initiator">
    <state>confirmed</state>
    <local>
      <target uri="sip:bob@example.com">
        <param pname="appearance" pvalue="0" />
        <param pname="+sip.rendering" pvalue="yes"/>
      </target>
    </local>
    <remote>
      <identity>sip:carol@example.com</identity>
      <target uri="sip:carol@example.com;user=phone" />
    </remote>
  </dialog>
</dialog-info>
```

F16 to F20: Bob places Carol on hold.

F22 to F24: Bob notifies Appearance Agent of the status of the dialog to indicate the held state. It indicates this by setting the sip.rendering parameter in the NOTIFY payload to (no). Appearance Agent notifies Alice of the same.

F22 Bob ----> Appearance Agent

NOTIFY sip:sa@stateagent.example.com SIP/2.0
Via: SIP/2.0/UDP ua2.example.com;branch=z9hG4bK6c78a6c5CA00520E
From: <sip:bob@example.com>;tag=44150CC6-A7B7919D
To: <sip:alice@example.com>;tag=428765950880801
CSeq: 10 NOTIFY
Call-ID: 144-1289338424@example.com
Contact: <sip:alice@ua2.example.com>
Event: dialog;ma
User-Agent: XYZ-UA/4.5.6

Subscription-State: active;expires=3338
Max-Forwards: 70
Content-Type: application/dialog-info+xml
Content-Length: 942

```
<?xml version="1.0"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
    version="9"
    state="partial"
    entity="sip:alice@example.com">
  <dialog id="id3d4f9c83"
    call-id="f3b3cbd0-a2c5775e-5df9f8d5@ua2.example.com"
    local-tag="15A3DE7C-9283203B"
    remote-tag="65a98f7c-1dd2-11b2-88c6-b03162323164+65a98f7c"
    direction="initiator">
    <state>confirmed</state>
    <local>
      <target uri="sip:bob@example.com">
        <param pname="appearance" pvalue="0" />
        <param pname="+sip.rendering" pvalue="no"/>
      </target>
    </local>
    <remote>
      <identity>sip:carol@example.com</identity>
      <target uri="sip:carol@example.com" />
    </remote>
  </dialog>
</dialog-info>
```

F26 to F34 : Alice picks up the held call by sending an INVITE with Replaces: header (F26). Session is established between Alice and Carol. The dialog between Carol and Bob is terminated.

F26 Alice ----> Proxy

```
INVITE sip:carol@example.com SIP/2.0
Via: SIP/2.0/UDP ua1.example.com;branch=z9hG4bK4ea695b5B376A60C
From: <sip:alice@example.com>;tag=8C4183CB-BCEAB710
To: <sip:carol@example.com:5075>
CSeq: 1 INVITE
Call-ID: 3d57cd17-47deb849-dca8b6c6@ua1.example.com
Contact: <sip:alice@ua1.example.com>
User-Agent: ABC-UA/1.2.3
P-Preferred-Identity: <sip:alice@example.com>
<all-one-line>
```


Replaces: f3b3cbd0-a2c5775e-5df9f8d5@ua2.example.com;to-tag=65a98f7c-1dd2-11b2-88c6-b03162323164+65a98f7c;from-tag=15A3DE7C-9283203B
</all-one-line>
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 223

v=0
o=- 1102980497 1102980497 IN IP4 ua1.example.com
s=IP SIP UA
c=IN IP4 ua1.example.com
t=0 0
a=sendrecv
m=audio 2238 RTP/AVP 0 8 101
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:101 telephone-event/8000

F34 to F41: Bob notifies the Appearance Agent of the termination of dialog at his UA. Alice notifies the Appearance Agent of the confirmed state of the dialog at her UA.

F34 Bob ----> Appearance Agent

NOTIFY sip:sa@stateagent.example.com SIP/2.0
Via: SIP/2.0/UDP ua2.example.com;branch=z9hG4bKa5d6cf61F5FBC05A
From: <sip:bob@example.com>;tag=44150CC6-A7B7919D
To: "State_Agent" <sip:alice@example.com>;tag=428765950880801
CSeq: 11 NOTIFY
Call-ID: 144-1289338424@example.com
Contact: <sip:alice@ua2.example.com>
Event: dialog;ma
User-Agent: XYZ-UA/4.5.6
Subscription-State: active;expires=3334
Max-Forwards: 70
Content-Type: application/dialog-info+xml
Content-Length: 677

```
<?xml version="1.0"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
  version="10"
  state="partial"
  entity="sip:alice@example.com:5060">
  <dialog id="id3d4f9c83"
    call-id="f3b3cbd0-a2c5775e-5df9f8d5@ua2.example.com"
    local-tag="15A3DE7C-9283203B"
    remote-tag="65a98f7c-1dd2-11b2-88c6-b03162323164+65a98f7c"
```



```
direction="initiator">
<state>terminated</state>
<local>
  <target uri="sip:bob@example.comsip:bob@example.com">
    <param pname="appearance" pvalue="0" />
  </target>
</local>
<remote>
  <identity>sip:carol@example.com</identity>
  <target uri="sip:carol@example.com" />
</remote>
</dialog>
</dialog-info>
```

F38 Alice ----> Appearance Agent

NOTIFY sip:sa@stateagent.example.com SIP/2.0
Via: SIP/2.0/UDP ua1.example.com;branch=z9hG4bK93f44af3518A1572
From: <sip:alice@example.com>;tag=5861255C-14C04045
To: "State_Agent" <sip:alice@example.com>;tag=920163082722420
CSeq: 10 NOTIFY
Call-ID: 143-1840952798@example.com
Contact: <sip:alice@ua1.example.com>
Event: dialog;ma
User-Agent: ABC-UA/1.2.3
Subscription-State: active;expires=3315
Max-Forwards: 70
Content-Type: application/dialog-info+xml
Content-Length: 640

```
<?xml version="1.0"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
  version="9"
  state="partial"
  entity="sip:alice@example.com:5060">
  <dialog id="id402f024e"
    call-id="3d57cd17-47deb849-dca8b6c6@ua1.example.com"
    local-tag="8C4183CB-BCEAB710"
    remote-tag="65a98f7c-1dd2-11b2-88c6-b03162323164+65a98f7c"
    direction="initiator">
    <state>confirmed</state>
    <local>
      <target uri="sip:alice@example.com">
        <param pname="appearance" pvalue="0" />
        <param pname="+sip.rendering" pvalue="yes"/>
      </target>
```



```
</local>
<remote>
  <identity>sip:carol@example.com</identity>
  <target uri="sip:carol@example.com" />
</remote>
</dialog>
</dialog-info>
```

F42 to F59: Carol terminates the dialog with Alice. Alice notifies the Appearance Agent of the dialog state (terminated). The Appearance Agent notifies Bob of the same.

F46 Alice ----> Appearance Agent

```
NOTIFY sip:sa@stateagent.example.com SIP/2.0
Via: SIP/2.0/UDP ua1.example.com;branch=z9hG4bKa46c2f85F29F839C
From: <sip:alice@example.com>;tag=5861255C-14C04045
To: "State_Agent" <sip:alice@example.com>;tag=920163082722420
CSeq: 11 NOTIFY
Call-ID: 143-1840952798@example.com
Contact: <sip:alice@ua1.example.com>
Event: dialog;ma
User-Agent: ABC-UA/1.2.3
Subscription-State: active;expires=3311
Max-Forwards: 70
Content-Type: application/dialog-info+xml
Content-Length: 642
```

```
<?xml version="1.0"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
  version="10"
  state="partial"
  entity="sip:alice@example.com">
  <dialog id="id402f024e"
    call-id="3d57cd17-47deb849-dca8b6c6@ua1.example.com"
    local-tag="8C4183CB-BCEAB710"
    remote-tag="65a98f7c-1dd2-11b2-88c6-b03162323164+65a98f7c"
    direction="initiator">
    <state>terminated</state>
    <local>
      <target uri="sip:alice@example.com">
        <param pname="appearance" pvalue="0" />
      </target>
    </local>
    <remote>
      <identity>sip:carol@example.com</identity>
      <target uri="sip:carol@example.com" />
```



```

    </remote>
  </dialog>
</dialog-info>

```

6.3. Call Offered to an Appearance Group

In the call flow below Bob has bridged appearance of Alice. Carol places a call to Alice. Both Alice and Bob's devices are alerted of the incoming call. Bob answers the call. He then places Carol on hold. Alice picks up the held call and has a established session with Carol. Finally, Carol terminates the session. All NOTIFY messages in the call flow below carry dialog events and only dialog states are mentioned for simplicity. For brevity, the details of some messages are not shown below.

Carol	Forking Proxy	Appearance Agent	Alice	Bob
>F1 INVITE >				
	>F2 INVITE ----->			
	>F3 INVITE ----->			
<-100Try F4<				
	<----- Ringing 180 F5<			
<180Ring F6<				
	<----- Ringing 180 F7<			
<180Ring F8<				
	<----- 200 OK F9<			
<-200OK F10<				
	>F11 CANCEL ----->			
	<----- 200 OK F12<			
	<Request Cancelled 487 F13<			
	>F14 ACK ----->			
>F15 ACK -->				
	>F16 ACK ----->			
<=====Both way RTP established=====>				
		<----- NOTIFY F17<		
		>F18 200 OK ----->		

		>F19 NOTIFY >	
		<- 2000K F20<	
	<-----("Hold") INVITE F21<		
<INVITE F22<			
>F23 2000K->			
	>F24 200 OK ----->		
	<----- ACK F25<		
<-- ACK F26<			
		<----- NOTIFY F27<	
		>F28 200 OK ----->	
		>F29 NOTIFY >	
		<- 2000K F30<	
	< INVITE (w/ Replaces) F31<		
<INVITE F32<			
(w/Replaces)			
>F33 200 OK>			
	>F34 200 OK ----->		
	<----- ACK F35<		
<-- ACK F36<			
<=====Both way RTP established=====>			
>F37 BYE -->			
	>F38 BYE ----->		
	<----- OK 200 F39<		
<-2000K F40<			
		<----- NOTIFY F41<	
		>F42 200 OK ----->	
		>F43 NOTIFY >	
		<-200 OK F44<	
		<-NOTIFY F45<	
		>F46 200 OK->	


```

|           |           |           |           | |
|           |           |>F47 NOTIFY ----->|
|           |           |           |           |
|           |           |<----- 200 OK F48<|
|>49 BYE --->|           |           |           |
|           |>F50 BYE ----->|           |           |
|           |           |           |           |
|           |<----- OK 200 F51<|           |           |
|<-200OK F52<|           |           |           |
|           |           |< NOTIFY F53<|           |           |
|           |           |           |           |
|           |           |>F54 200 OK >|           |           |
|           |           |           |           |
|           |           |>F55 NOTIFY ----->|           |           |
|           |           |           |           |
|           |           |<----- 200 OK F56<|           |           |
|           |           |           |           |

```

F1 to F16: An incoming call from Carol to Alice is forked to Bob and Alice. Both Alice and Bob indicate an incoming call (e.g., ringing) from Carol. Bob answers the call and two-way media is established between Carol and Bob.

F2 Proxy ----> Bob

```

INVITE sip:alice@ua3.example.com SIP/2.0
Via: SIP/2.0/UDP ua3.example.com;branch=z9hG4bK4324ea695b5B376A
Via: SIP/2.0/UDP proxy.example.com;branch=z9hG4bK38432ji
From: <sip:carol@example.com>;tag=94183CB-BCEAB7
To: <sip:alice@example.com>
CSeq: 106 INVITE
Call-ID: 47deb849-dca8b6c6-3d342
Contact: <sip:carol@ua3.example.com>
Max-Forwards: 69
Alert-Info: <file://ring.pcm>;alert=normal;appearance=0
Content-Type: application/sdp
Content-Length: 223

```

```

v=0
o=- 1102980499 1102980499 IN IP4 ua3.example.com
s=
c=IN IP4 ua3.example.com
t=0 0
a=sendrecv
m=audio 2238 RTP/AVP 0 8 101
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000

```


a=rtpmap:101 telephone-event/8000

F3 Proxy ----> Alice

```
INVITE sip:alice@ua1.example.com SIP/2.0
Via: SIP/2.0/UDP ua3.example.com;branch=z9hG4bK4324ea695b5B376A
Via: SIP/2.0/UDP proxy.example.com;branch=z9hG4bK348281
From: <sip:carol@example.com>;tag=94183CB-BCEAB7
To: <sip:alice@example.com>
CSeq: 106 INVITE
Call-ID: 47deb849-dca8b6c6-3d342
Contact: <sip:carol@ua3.example.com>
Max-Forwards: 69
Alert-Info: <file://ring.pcm>;alert=normal;appearance=0
Content-Type: application/sdp
Content-Length: 223
```

```
v=0
o=- 1102980499 1102980499 IN IP4 ua3.example.com
s=
c=IN IP4 ua3.example.com
t=0 0
a=sendrecv
m=audio 2238 RTP/AVP 0 8 101
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:101 telephone-event/8000
```

F17 - F20: Bob notifies the Appearance Agent with dialog state payload indicating the dialog in confirmed state. Appearance Agent notifies Alice of the status of the dialog at Bob.

F17 Bob ----> Appearance Agent

```
NOTIFY sip:sa@stateagent.example.com SIP/2.0
Via: SIP/2.0/UDP ua2.example.com;branch=z9hG4bK58a0dd68C2D63263
From: <sip:bob@example.com>;tag=558C18F7-DB9DF7BC
To: <sip:alice@example.com>;tag=1894685100249086
CSeq: 14 NOTIFY
Call-ID: 77-505889516@example.com
Contact: <sip:alice@ua2.example.com>
Event: dialog;ma
User-Agent: XYZ-UA/4.5.6
Subscription-State: active;expires=3427
Max-Forwards: 70
```


Content-Type: application/dialog-info+xml

Content-Length: 575

```
<?xml version="1.0"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
  version="13"
  state="partial"
  entity="sip:alice@example.com">
  <dialog id="ida0f8dc17"
    call-id="14-1541707345@example.com"
    local-tag="44BAD75D-E3128D42"
    remote-tag="d3b06488-1dd1-11b2-88c5-b03162323164+d3e48f4c"
    direction="recipient">
    <state>confirmed</state>
    <local>
      <target uri="sip:alice@example.com">
        <param pname="appearance" pvalue="0" />
        <param pname="+sip.rendering" pvalue="yes"/>
      </target>
    </local>
    <remote>
      <identity>sip:carol@ua.example.com</identity>
    </remote>
  </dialog>
</dialog-info>
```

F19 Appearance Agent ----> Alice

NOTIFY sip:alice@ua1.example.com SIP/2.0
From: <sip:alice@example.com>;tag=151702541050937
To: <sip:alice@example.com>;tag=18433323-C3D237CE
Call-ID: 1e361d2f-a9f51109-bafe31d4@ua1.example.com
CSeq: 12 NOTIFY
Via: SIP/2.0/UDP stateagent.example.com;branch=z9hG4bK14031499568413
Max-Forwards: 70
Content-Type: application/dialog-info+xml
Event: dialog;ma
Subscription-State: active
Contact: <sip:sa@stateagent.example.com>
Content-Length: 618

```
<?xml version="1.0"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
  version="13"
  state="partial"
  entity="sip:alice@example.com">
  <dialog id="2a7294823093f5274e3fd2ec54a2d76c"
```



```
call-id="14-1541707345@example.com"
local-tag="44BAD75D-E3128D42"
remote-tag="d3b06488-1dd1-11b2-88c5-b03162323164+d3e48f4c"
direction="recipient">
<state>confirmed</state>
  <local>
    <target uri="sip:alice@example.com">
      <param pname="appearance" pvalue="0"/>
      <param pname="+sip.rendering" pvalue="yes"/>
    </target>
  </local>
<remote>
  <identity>sip:carol@ua.example.com</identity>
</remote>
</dialog>
</dialog-info>
```

F21 to F26: Bob places Carol on hold.

F27 to F30: Bob notifies the Appearance Agent of the status of the dialog on hold with inclusion of the session description in the NOTIFY payload. Subsequently, Appearance Agent notifies Alice of the status of dialog.

F31 to F40: Alice picks up the held call by sending an INVITE with Replaces: header populated with the dialog data received in the NOTIFY from the Appearance Agent. Carol establishes a session with Alice and terminates the dialog with Bob.

F31 Alice ----> Proxy

```
INVITE sip:carol@ua.example.com SIP/2.0
Via: SIP/2.0/UDP ua1.example.com;branch=z9hG4bKcc9d727c2C29BE31
From: <sip:alice@example.com>;tag=605AD957-1F6305C2
To: <sip:carol@ua.example.com>
CSeq: 2 INVITE
Call-ID: dc95da63-60db1abd-d5a74b48@ua1.example.com
Contact: <sip:alice@ua1.example.com>
User-Agent: ABC-UA/1.2.3
P-Preferred-Identity: <sip:alice@example.com>
<all-one-line>
Replaces: 14-1541707345@example.com;to-tag=d3b06488-1dd1-11b2
-88c5-b03162323164+d3e48f4c;from-tag=44BAD75D-E3128D42
</all-one-line>
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 223
```



```

v=0
o=- 1103061265 1103061265 IN IP4 ua1.example.com
s=IP SIP UA
c=IN IP4 ua1.example.com
t=0 0
a=sendrecv
m=audio 2236 RTP/AVP 0 8 101
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:101 telephone-event/8000

```

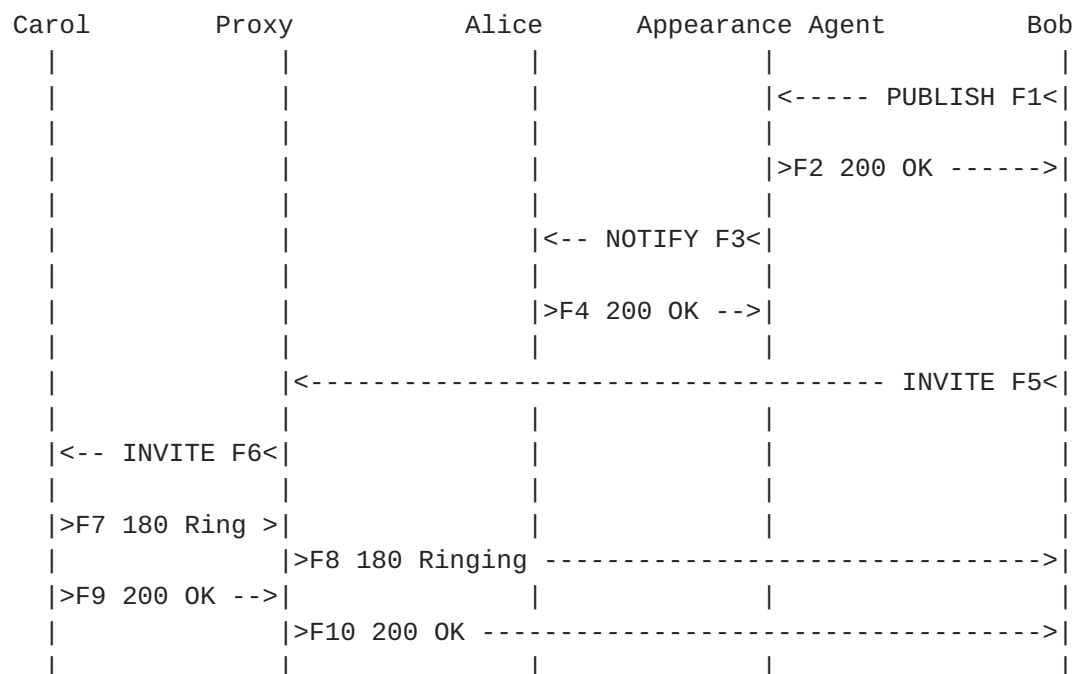
F41 to F48: Bob notifies the Appearance Agent of the termination of the dialog and Appearance Agent notifies the same to Alice. Alice notifies the Appearance Agent of the confirmed state of dialog at Alice and Appearance Agent informs Bob of the same.

F49 to F52: Carol terminates dialog with Alice.

F53 to F56: Alice notifies the Appearance Agent of the termination of the dialog and Appearance Agent notifies Bob of the same.

[6.4.](#) Use of PUBLISH

This call flow shows the use of PUBLISH instead of SUBSCRIBE/NOTIFY between the members of the appearance group and the Appearance Agent.




```

|<----- ACK F11<|
|
|
|<===== Both way RTP established =====>|
|
|
|
|<----- PUBLISH F12<|
|
|
|>F13 200 OK ----->|
|
|<- NOTIFY F14<|
|
|>F15 200 OK ->|
|
|
|<----- (hold) INVITE F16<|
|<- INVITE F17<|
|
|>F18 200 OK ->|
|
|>F19 200 OK ----->|
|
|
|<----- ACK F20<|
|
|
|<----- PUBLISH F21<|
|
|
|>F22 200 OK ----->|
|
|<- NOTIFY F23<|
|
|>F24 200 OK ->|
|
|<-- INVITE F25<|
|<- INVITE F26<|(w/ Replaces)
| ( w/ Replaces)|
|>F27 200 OK ->|
|
|>F28 200 OK -->|
|
|<----- ACK F29<|
|
|<= Both way RTP established =>|
|
|>F30 BYE ----->|
|
|>F31 BYE ----->|
|
|
|<----- OK 200 F32<|
|<- 200 OK F33<|
|
|
|<----- PUBLISH F34<|
|
|
|>F35 200 OK ----->|
|
|<- NOTIFY F36<|
|
|>F37 200 OK ->|
|
|

```



```

|                                     |>F38 PUBLISH >|
|                                     |
|                                     |<- 200 OK F39<|
|                                     |>F40 NOTIFY ----->|
|                                     |<----- 200 OK F41<|
|>F42 BYE ----->|
|>F43 BYE ----->|
|<-- 200 OK F44<|
|<--200 OK F45<|
|                                     |>F46 PUBLISH >|
|                                     |<- 200 OK F47<|
|                                     |>F48 NOTIFY ----->|
|                                     |<----- OK 200 F49<|

```

6.5. Bridging on an Appearance

In this call flow, a call answered by Bob is joined by Alice or "bridged". The Join header field is used by Alice to request this bridging. If Bob did not support media mixing, Bob could obtain conferencing resources as described in [12].

Carol	Forking Proxy	Appearance Agent	Alice	Bob
>F1 INVITE >				
	>F2 INVITE ----->			
	>F3 INVITE ----->			
<-100Try F4<				
	<----- Ringing 180 F5<			
<180Ring F6<				
	<----- Ringing 180 F7<			
<180Ring F8<				
	<----- 200 OK F9<			
<-200OK F10<				
	>F11 CANCEL ----->			
	<----- 200 OK F12<			
	<Request Cancelled 487 F13<			


```

|                                     |
|>F14 ACK ----->|
|>F15 ACK -->|
|                                     |
|>F16 ACK ----->|
|                                     |
|<=====Both way RTP established=====>|
|                                     |
|                                     |<----- NOTIFY F17<|
|                                     |
|                                     |>F18 200 OK ----->|
|                                     |
|                                     |>F19 NOTIFY >|
|                                     |
|                                     |<- 200OK F20<|
|                                     |
|<---- INVITE (w/ Join) F21<|
|                                     |
|>F22 INVITE (w/Join)----->|
|                                     |
|<---- OK 200 Contact:Bob;isfocus F23<|
|                                     |
|>F24 200 OK Contact:Bob;isfocus----->|
|                                     |
|<----- ACK F25<|
|                                     |
|>ACK F26----->|
|                                     |
|<-----INVITE Contact:Bob;isfocus F27<|
|<-INVITE F28|
|>F29 200 -->|
|                                     |
|>F30 200 OK ----->|
|                                     |
|<----- ACK F31<|
|<--- ACK F32|
|                                     |<==RTP==>|
|<=====Both way RTP established=====>|

```

F21 Alice -----> Proxy

```

INVITE sip:bob@ua.example.com SIP/2.0
Via: SIP/2.0/UDP ua1.example.com;branch=z9hG4bKcc9d727c2C29BE31
From: <sip:alice@example.com>;tag=605AD957-1F6305C2
To: <sip:bob@ua.example.com>
CSeq: 2 INVITE
Call-ID: dc95da63-60db1abd-d5a74b48@ua1.example.com
Contact: <sip:alice@ua1.example.com>
User-Agent: ABC-UA/1.2.3
P-Preferred-Identity: <sip:alice@example.com>

```



```

<all-one-line>
Join: 14-1541707345@example.com;to-tag=d3b06488-1dd1-11b2-88c5
-b03162323164+d3e48f4c;from-tag=44BAD75D-E3128D42
</all-one-line>
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 223

v=0
o=- 1103061265 1103061265 IN IP4 ua1.example.com
s=IP SIP UA
c=IN IP4 ua1.example.com
t=0 0
a=sendrecv
m=audio 2236 RTP/AVP 0 8 101
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:101 telephone-event/8000

```

6.6. State and Error Recovery Examples

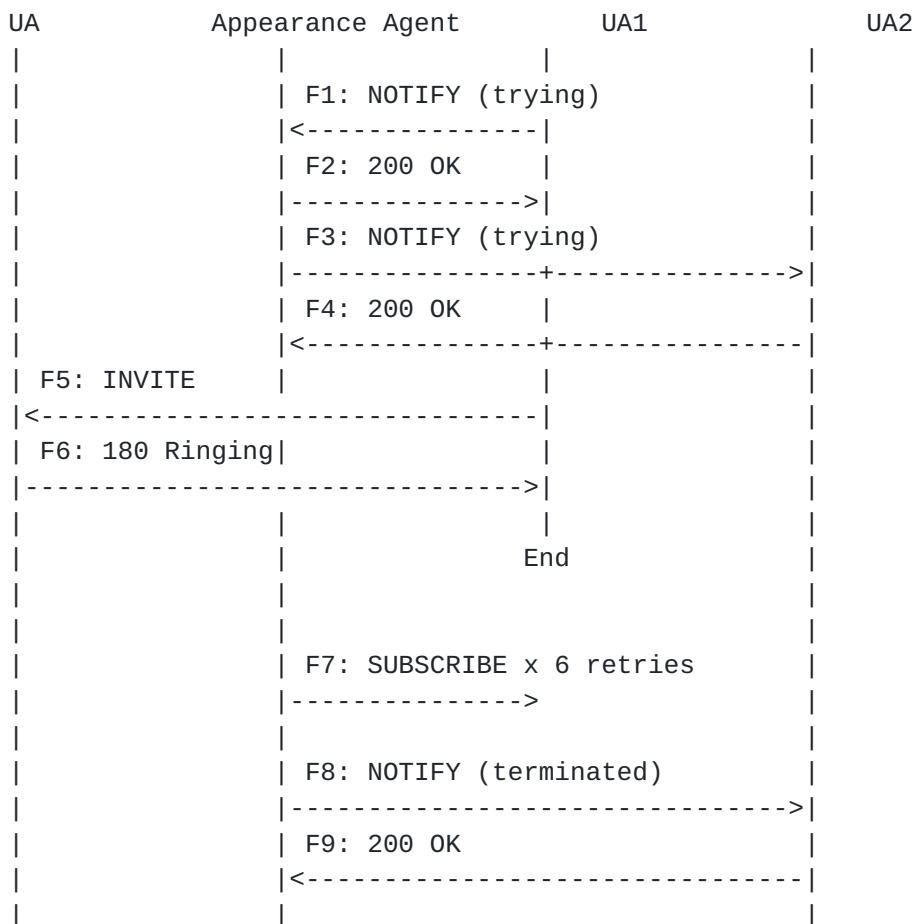
6.6.1. Line Seize (Refresh Subscription)

UA - Called	Appearance Agent	UA - Calling
	F1: NOTIFY (trying)	
	<-----	
	F2: 200 OK	
	----->	
F3: INVITE/180 Ring/200 OK/ACK		
<-----+-----		
	F4: SUBSCRIBE	
	----->	
	F5: 200 OK	
	<-----	
	F6: NOTIFY(confirm)	
	<-----	
	F7: 200 OK	
	----->	

This figure shows UA seizing a bridged line (F1 and F2) from the Appearance Agent. Appearance Agent refreshes its subscription to UA (F4-F7) ensuring continuity of service (whilst also verifying User agents shared line service). UA should maintain a policy of

shortened expires periods so long as it holds a line seize (throughout the period of a call). Subscription refreshes will therefore continue to use a shortened expires period. Although UA will determine the expiration period of subscriptions to it, the Appearance Agent may choose to refresh subscriptions on a more regular basis as an extra means of ensuring continuity of shared line service.

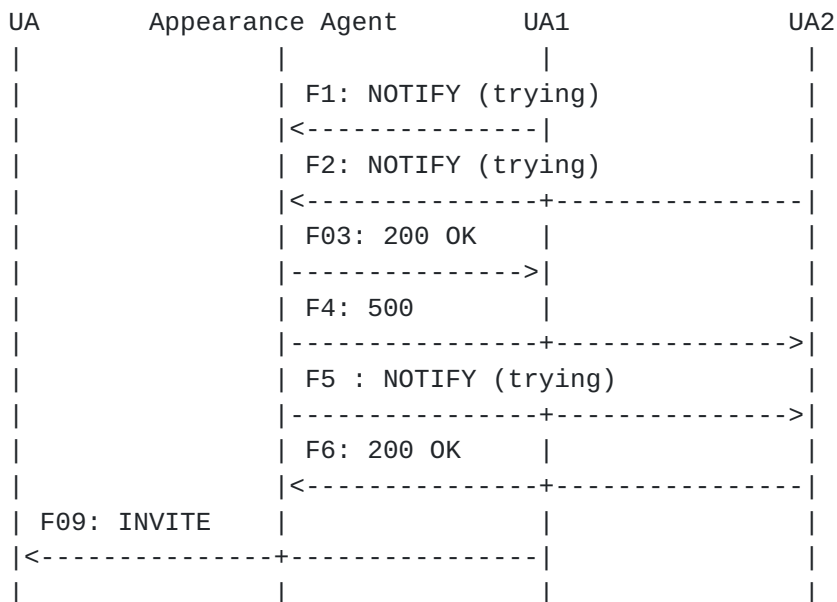
6.6.2. Line Seize (Notifier Failure)



The flow shown in this figure illustrates the failure of a user agent after it has obtained a line seize (F1-F2). Messages used to refresh the subscription from Appearance Agent to UA1 are shown at F7. The discontinuation of the bridged line service within user agent UA1 is shown by the abrupt termination of the UA1 vertical time line. When the Appearance Agent attempts to refresh its subscription and no response is received, indicating the shared line service maintained by UA1 has failed. Appearance Agent should at this point free the seize lock held by UA1 and issue NOTIFY messages (F8) indicating the

termination of the dialog associated with the shared line.

6.6.3. Line Seize (Race Condition)



This figure illustrates two user agents, UA1 and UA2, attempting to seize the same bridged line simultaneously. This type of race condition is often referred as a glare condition. Appearance Agent provides only one of UA1 and UA2 with the initiator's line seize (UA1 in this case) and may choose any policy deemed appropriate to resolve the race.

7. IANA Considerations

This section registers the SIP Alert-Info header field parameter "appearance" and the XML namespace extensions to the SIP Dialog Package.

The namespace URIs for the elements defined by this specification are URNs [RFC 4211], using the namespace identifier 'ietf' defined by [4] and extended by [6]:

```
urn:ietf:params:xml:ns:dialog-info:ma
```

This specification also defines a new event parameter "ma" for the Dialog Package.

8. Security Considerations

Since many of these features are implemented using semantics provided by [RFC 3261](#) [2], Event Package for Dialog State as define in [7], and Event Notification [4], security considerations in these documents apply to this draft as well.

Specifically, since dialog state information and the dialog identifiers are supplied by UA's in an appearance group to other members, the same is prone to "call hijacks". For example, a rogue UA could snoop for these identifiers and send an INVITE with Replaces header containing these call details to take over the call. As such INVITES with Replaces header MUST be authenticated using the standard mechanism (like Digest or S/MIME) described in [RFC 3261](#) [2] before it is accepted. NOTIFY message bodies that provide the dialog state information and the dialog identifiers MAY be encrypted end-to-end using the standard mechanics. All SUBSCRIBES between the UA's and the Event Agent MUST be authenticated.

9. Acknowledgements

The authors would like to thank Kent Fritz, John Weald, and Sunil Veluvali of Sylanro Systems, Steve Towlson, and Michael Procter of Citel Technologies, Rob Harder and Hong Chen of Polycom Inc, John Elwell, J D Smith of Siemens Communications, Dale R. Worley of Pingtel, and Graeme Dollar of Yahoo Inc, for their numerous corrections, comments and suggestions during authoring of this draft.

10. Normative References

- [1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [2] 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.
- [3] Sparks, R., "The Session Initiation Protocol (SIP) Refer Method", [RFC 3515](#), April 2003.
- [4] Roach, A., "Session Initiation Protocol (SIP)-Specific Event Notification", [RFC 3265](#), June 2002.
- [5] Mahy, R., Biggs, B., and R. Dean, "The Session Initiation Protocol (SIP) "Replaces" Header", [RFC 3891](#), September 2004.

- [6] Johnston, A., "Session Initiation Protocol Service Examples", [draft-ietf-sipping-service-examples-12](#) (work in progress), January 2007.
- [7] Rosenberg, J., Schulzrinne, H., and R. Mahy, "An INVITE-Initiated Dialog Event Package for the Session Initiation Protocol (SIP)", [RFC 4235](#), November 2005.
- [8] Rosenberg, J., "A Session Initiation Protocol (SIP) Event Package for Registrations", [RFC 3680](#), March 2004.
- [9] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", [RFC 3264](#), June 2002.
- [10] Jennings, C., Peterson, J., and M. Watson, "Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks", [RFC 3325](#), November 2002.
- [11] Mahy, R. and D. Petrie, "The Session Initiation Protocol (SIP) "Join" Header", [RFC 3911](#), October 2004.
- [12] Johnston, A. and O. Levin, "Session Initiation Protocol (SIP) Call Control - Conferencing for User Agents", [BCP 119](#), [RFC 4579](#), August 2006.
- [13] Johnston, A., "Requirements and Implementation Options for the Multiple Line Appearance Feature using the Session Initiation Protocol (SIP)", [draft-johnston-bliss-mla-req-00](#) (work in progress), February 2007.
- [14] Niemi, A., "Session Initiation Protocol (SIP) Extension for Event State Publication", [RFC 3903](#), October 2004.

Authors' Addresses

Mohsen Soroushnejad (editor)
Sylantro Systems Corp

Email: mohsen.soroush@sylantro.com

Venkatesh Venkataramanan
Sylantro Systems Corp

Email: venkatar@sylantro.com

Paul Pepper
Citel Technologies

Email: paul.pepper@citel.com

Anil Kumar
Yahoo Inc.

Email: anil@yahoo-inc.com

Alan Johnston (editor)
Avaya
St. Louis, MO 63124

Email: alan@sipstation.com

Full Copyright Statement

Copyright (C) The IETF Trust (2007).

This document is subject to the rights, licenses and restrictions contained in [BCP 78](#), and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY, THE IETF TRUST AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM 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.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in [BCP 78](#) and [BCP 79](#).

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at <http://www.ietf.org/ipr>.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).

