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 August 18, 2008.

Copyright Notice

Copyright (C) The IETF Trust (2008).

Abstract

This document analyzes the interoperability problems surrounding the call-completion feature that allows a callee to put a caller's request into a queue by which the caller can be notified to call back the callee at later time. This document analyzes how different solutions inter-operate and tries to make recommendation on how to best meet this requirement.
Table of Contents

1. Introduction ................................................. 3
2. Requirements Terminology ..................................... 3
3. Terminology .................................................. 3
4. Overview ..................................................... 4
5. Detailed Description of the Call-Completion Mechanism ........ 5
   5.1. Caller’s Call-Completion Agent ........................... 5
   5.2. Callee’s Call-Completion Monitor ......................... 6
   5.3. The Original Call Is Made ................................ 6
   5.4. Call-Completion Is Invoked ............................... 7
   5.5. The Call-Completion Request Is Queued .................... 8
   5.6. Call-Completion Is Activated ............................. 8
   5.7. Data Provided in the Call-Completion Event Package ...... 10
6. Call Completion Event Package .................................. 10
   6.1. Event Package Name ....................................... 10
   6.2. Event Package Parameters ................................ 10
   6.3. SUBSCRIBE Bodies ........................................ 11
   6.4. Subscribe Duration ....................................... 11
   6.5. NOTIFY Bodies ............................................ 11
   6.6. Subscriber Generation of SUBSCRIBE Requests ............ 12
   6.7. Notifier Processing of SUBSCRIBE Requests ............... 12
   6.8. Notifier Generation of NOTIFY Requests .................. 12
   6.9. Subscriber Processing of NOTIFY Requests ................ 13
   6.10. Handling of Forked Requests ............................ 13
   6.11. Rate of Notifications ................................... 13
7. Security Considerations ........................................ 13
8. IANA Considerations ........................................... 14
9. References .................................................... 14
   9.1. Normative References ..................................... 14
   9.2. Informative References ................................... 15
Authors’ Addresses .............................................. 15
Intellectual Property and Copyright Statements ................. 16
1. Introduction

The call-completion architecture is driven by the interactions between two types of agents, a "caller’s agent" which operates on behalf of the original caller, and a "callee’s monitor", which operates on behalf of the original callee. In order to allow flexibility and innovation, most of the interaction between the caller’s agent and the caller-user(s) and the caller’s UA(s) is out of the scope of this document.

Similarly, most of the interaction between the callee’s monitor and the callee-user(s) and the callee’s UA(s) is out of the scope of this document, as is also the policy by which the callee’s monitor arbitrates between multiple call-completion requests. (Although simple agents and monitors can be devised that interact with users and UAs entirely through standard SIP mechanisms[RFC3265][RFC4235][RFC3515].)

2. Requirements Terminology

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 [RFC2119].

3. Terminology

For the purpose of this service, we provide the following terminologies:

CCBS: Completion of Calls to Busy Subscribers

CCNR: Completion of Calls on No Reply

CCBS/CCNR service duration timer: maximum time the CCBS/CCNR request will remain activated for the caller within the network.

Call-completion call: a call from the call-completion user to the call-completion target, triggered as a result of the execution of a call completion service.

Call-completion queue: a buffer at the callee which queues automatically not answered calls due to busy or no reply.

Callee: called user, busy when the first call arrives, target of the call-completion call.
Callee Monitor:

Caller Agent:

Caller: calling user, encounters a busy callee at the first call, initiator of the call-completion call.

Retain option: a characteristic of the call-completion service; if supported, call-completion calls which again encounter a busy callee will not be queued again, but the position of the caller’s entry in the queue is retained.

4. Overview

The call-completion architecture augments each caller’s UA (or UAC) which wishes to be able to use the call-completion features with with a "call-completion agent" (also written as "CC agent", "agent", or "caller’s agent"). It augments each callee’s UA (or UAS) which wishes to be able to be the target of the call-completion features with a "call-completion monitor" (also written as "CC monitor", "monitor", or "callee’s monitor"). The caller’s agent subscribes to the call-completion event package[RFC3265] of the callee’s monitor in order to coordinate with the monitor (and indirectly with other callee’s monitors) to implement the call-completion features.

When the caller’s UA makes a call to a callee that fails (e.g., because the callee was busy or the callee did not answer), and the caller wishes to use CC to contact the callee later, the caller instructs the caller’s agent to activate the CC feature.

Note that SIP call-completion does not inherently distinguish "call completion on no reply" (CCNR) from "call completion on busy subscriber" (CCBS), because the network does not need to make a distinction, and given the potential complexity of SIP routing, agents in the network may not be able to.

The caller’s agent sends a SUBSCRIBE request for the call-completion event package to the original destination URI of the call. This SUBSCRIBE reaches the callee’s monitor. The callee’s monitor uses the existence of the subscription to know that the caller is interested in using the CC feature in regard to the original call. The monitor keeps a list or queue of failed calls to the callee, and of the caller’s agent subscriptions, which indicate the callers that are waiting to use the CC features.

When the callee’s monitor judges that the user and/or user’s UA is available for call-completion, the callee’s monitor selects (usually)
one caller’s agent to be the next caller to execute call-completion to the callee. The callee’s monitor sends a call-completion event update to the selected caller’s agent telling it to begin execution of call-completion.

When the caller’s agent receives this update, it calls the caller’s UA or otherwise tests whether the caller is available to take advantage of call-completion. If the caller is available, the agent directs the caller’s UA to make again the call to the callee. This call is identified as a call-completion call so it can be given precedence in reaching the callee’s UA.

5. Detailed Description of the Call-Completion Mechanism

5.1. Caller’s Call-Completion Agent

The call-completion architecture augments each caller’s UA (or UAC) which wishes to be able to use the call-completion features with a “call-completion agent”. An agent may be associated with more than one UA if it is common that a caller or population of users will be shared between the UAs, and especially if the UAs share an AOR.

The caller’s agent has a method of monitoring calls made from the UA(s) in order to determine their Call-Id’s and (potentially) their final response statuses. This may be achieved by subscribing to the dialog event package of the UA(s) or by other means.

The callers using the UA(s) can indicate to the caller’s agent when they wish to avail themselves of CC for a recently-made call which failed to reach their chosen destination. This may be achieved by an INVITE to a special URI which is routed to the caller’s agent or by other means.

The caller’s agent has a method of monitoring the status of the UA(s) to determine when they are available to be used for a CC call. This may be achieved by subscribing to the dialog event package of the UA(s) or by other means.

The caller’s agent can communicate to the UA(s) that CC has become active and to inquire if the relevant calling user is available for the CC call. This may be achieved by sending an INVITE to the UA(s) or by other means.

The caller’s agent can order the UA(s) at which the relevant calling user is available to generate a CC call to the callee. This may be achieved by sending a REFER to the UA(s) or by other means.
5.2. Callee’s Call-Completion Monitor

The call-completion architecture augments callee’s UA (or UAS) which wishes to be able to be the target of the call-completion features with a "call-completion monitor". A monitor may be associated with more than one UA if it is common that a callee or population of users will be shared between the UAs, and especially if the UAs share an AOR.

The callee’s monitor has a method of monitoring calls made to the UA(s) in order to determine their Call-Id’s and (potentially) their final response statuses. This may be achieved by subscribing to the dialog event package of the UA(s) or by other means (e.g., by communication with the UA’s "home proxy").

The callee’s using the UA(s) may be able to indicate to the callee’s monitor when they wish to receive CC calls. This may be achieved by an INVITE to a special URI which is routed to the callee’s monitor or by other means.

The callee’s monitor has a method of monitoring the status of the UA(s) to determine when they are in a suitable state to receive a CC call. This state may vary depending on the type of CC call in question. E.g., a UA is available for CCBS when it is not busy, but a UA is available for CCNR when it becomes not busy after being busy with an established call. This monitoring may be achieved by subscribing to the dialog event package of the UA(s) or by other means.

The callee’s monitor maintains information about the set of INVITEs that have been received by the UA(s) that did not obtain successful final responses. In practice, the monitor may remove knowledge about an incoming dialog from its set if its CC policy establishes that the dialog is no longer eligible for CC.

5.3. The Original Call Is Made

The caller’s UA sends an INVITE to a request URI. One or more forks of this request reach one or more of the callee’s UAs. By hypothesis, none of the callee’s UAs returns a success response, as otherwise, call completion services would not be needed for this call. However, the caller’s INVITE might succeed at some other UA that the calling user considers insufficient to satisfy his needs. E.g., a call that is not answered by the callee user may connect to the callee user’s voicemail server. Eventually, the INVITE fails, or the resulting dialog(s) are terminated. Note that the Call-Id of the INVITE is a unique identifier of this call attempt.
The caller’s agent records the request URI, the Call-Id, and possibly the final request status that the caller’s UA received. The callee’s monitor records the Call-Id and possibly the final request status(es) returned by the callee’s UA(s).

Note that the caller’s UA may not receive any response from any of the callee’s UA(s), as the final response returned to the caller’s UA may have been from a fork that reached a UA that was not the callee’s.

### 5.4. Call-Completion Is Invoked

The calling user indicates to the caller’s agent that he wishes to invoke call-completion services on the recent call. Note that from the SIP point of view, the INVITE may be successful, but from the user’s point of view, the call may be unsuccessful. E.g., the call may have connected to the callee’s voicemail, which would return a 200 status to the INVITE but from the caller’s point of view is "no reply".

Question: At this point, it seems that the best choice is that the caller’s agent need not determine what type of CC is being requested (CCNR vs. CCBS), as (1) it cannot determine this from the INVITE final response, (2) it would be a burden to make the calling user to specify it, and (3) the callee’s monitor can determine this from the responses returned by the callee’s UAs.

The caller’s agent subscribes to the call-completion event package using the request URI of the original call. This SUBSCRIBE should be routed in much the same way as the original INVITE, but ultimately being routed not to the callee’s UAs but to the callee’s monitor. The Event header of the subscribe specifies the call-completion event package with a parameter call_id={Call-Id of the original call}.

Question: Should the specification of the original call be done in the SUBSCRIBE body rather than in an event-param?

The SUBSCRIBE should have headers to optimize its routing. In particular, it should contain "Request-Disposition: parallel, no-cancel", and an Accept-Contact header to eliminate callee UAs that are not acceptable to the calling user.

The callee’s monitor(s) that receive the SUBSCRIBE establish subscriptions. These subscriptions represent the caller’s agent’s request for call-completion services. The callee’s monitor must be prepared to receive multiple forks of a single SUBSCRIBE, and should respond 482 (Merged Request) to all but one fork. The callee’s monitor must be prepared to receive SUBSCRIBE$s regarding original
calls that it has no knowledge of, and should respond 404 (Not Found) to such SUBSCRIBEES. The monitor may apply additional restrictions as to which caller’s agents may subscribe.

The caller’s agent must be prepared to receive multiple responses to the SUBSCRIBE and to have multiple subscriptions established. The agent must also be prepared to have the SUBSCRIBE fail, in which case, CC cannot be invoked for this original call.

The call-completion event package returns various information to the caller’s agent, but the vital datum is that it contains an indication whether the callee’s monitor has chosen the caller’s agent to perform the next CC call to the callee. This datum is initially false.

5.5. The Call-Completion Request Is Queued

The continuation of the caller’s agent’s subscription indicates that the caller’s agent is prepared to initiate the CC call when it is selected by the callee’s monitor. If the caller’s agent becomes unwilling to initiate the CC call (e.g., because the calling user has deactivated CC or because the caller’s UA becomes busy), the caller’s agent must terminate or suspend the subscription(s). (Currently, no method of suspending a subscription is defined.) If the caller’s agent later becomes willing again to initiate CC for the original call, it may resume the suspended subscription(s) or initiate new one(s).

If the callee’s monitor becomes aware that, according to its policy, the original call referenced by a subscription will never be selected for call-completion, it should terminate the subscription. (And respond to any attempt to start a new subscription for that original call with 404.)

5.6. Call-Completion Is Activated

The callee’s monitor has a policy regarding when and how it selects CC requests to be activated. This policy may take into account the type of the requests (CCNR vs. CCBS), the state of the callee’s UA(s), the order in which the original calls arrived, and any previous CC attempts for the same original call. Usually the callee’s monitor will choose only one CC request for activation at a time, but if the callee’s UA(s) can support multiple calls, it may choose more than one.

The callee’s monitor changes the "call completion active" datum for the chosen caller’s agent from false to true. This triggers a notification for the agent’s subscription.
The agent receives the notification with the CC active datum set to true. It then terminates or suspends all other CC subscriptions for this original call, and all CC subscriptions for all other original calls, in order to prevent any other CC requests from this caller from being activated. The agent then determines whether the calling user is available for the CC call, usually by calling the caller’s UA(s).

If the calling user is not available, the caller’s agent indicates this to the callee’s monitor by terminating the CC subscription.

If the calling user is available, the caller’s agent causes the caller’s UA to initiate a call to the request URI (which is expected to be routed to the callee’s UA(s)).

Question: Should the callee’s monitor supply a URI which should be used in the CC call? This seems like it would be more reliable, as the monitor is probably "for" a particular callee URI, and it has no information about the destinations of any other forks of the original call.

Question: The CC must be marked in some way as a CC call in order for the callee’s monitor to know that the CC activation is being acted upon by the caller’s agent. And the marking must include the original Call-Id to allow correlation with the original call. Possibilities for a marking are a special URI-parameter on the request URI or a special header.

The callee’s UA(s) and any associated proxies may give the CC call precedence over non-CC calls.

The callee’s monitor supervises the receiving of the CC call. If the CC call does not arrive at the callee’s UA(s) promptly, the monitor will withdraw CC activation from the caller’s agent by changing the value of its CC active datum to false. Similarly, if the CC call fails, the monitor will withdraw CC activation. Depending on its policy, the same original call may be selected again for CC activation at a later time. If the CC call succeeds, the monitor will also withdraw CC activation, but the original call will never again be selected for CC activation (and in practice, can be deleted from the monitor’s records).

Question: Is that last statement true? Can a call appear to succeed from the monitor’s point of view but fail from the calling user’s point of view?

Once the CC call has failed, or if it has succeeded, once the CC call has been terminated, the callee’s monitor’s policy may select another
CC request for activation.

5.7. Data Provided in the Call-Completion Event Package

Question: What format should the event package data be presented in? This draft proposes a simple attribute-value format. We might also consider yet another XML format.

The only necessary information to be provided by the call-completion event package is the CC activation datum, whose value is false (meaning that this CC request has not been chosen for activation) or true (meaning that it has).

Question: If we decide to let the callee’s monitor provide the request URI for the CC call, that request URI should probably be a mandatory datum as well.

The event package may provide information about the callee’s monitor’s policy. In particular, the PSTN CC feature gives an indication of the "service retention" attribute, which indicates whether the CC request can be continued to a later time if the call-completion call fails due to the callee’s UA(s) being busy.

If the callee has a caller-queuing facility, we want to treat the call-completion queue as part of the queuing facility, and include in the event package information regarding the state of the queue, such as number of callers ahead of this caller and expected wait time. In that case, this data should probably not trigger a notification every time it changes, but rather at suitable time increments.

6. Call Completion Event Package

This section fills in the details needed to specify a possible call-completion event package, in accordance with section 4.4 of [RFC3265].

6.1. Event Package Name

The SIP Events specification requires package definitions to specify the name of their package or template-package. The name of this package is "call-completion". As specified in [RFC3265], this value appears in the Event and Allow-events header fields.

6.2. Event Package Parameters

No package specific Event header parameters are defined for this event package.
6.3. SUBSCRIBE Bodies

[RFC3265] requires package definitions to define the usage, if any, of bodies in SUBSCRIBE requests. A SUBSCRIBE request for a call-completion package MAY contain a body. This body defines a filter to be applied to the subscription. Filter documents are not specified in this document.

The SUBSCRIBE request MAY contain an Accept header field. If no such header field is present, it has a default value of "application/call-completion". If the header field is present, it MUST include "application/call-completion".

6.4. Subscribe Duration

[RFC3265] requires package definitions to define a default value for subscription durations, and to discuss reasonable choices for durations when they are explicitly specified.

It is recommended to set the default duration of subscriptions to call completion events to a value higher than 3600 seconds which corresponds to the highest timer value recommended for the call completion services in ETSI and ITU-T. The duration of the subscription is also coupled to the remaining duration of a queue entry. This means in case of resuming a subscription the resulting duration will be less than 3600 seconds.

6.5. NOTIFY Bodies

[RFC3265] requires package definitions to describe the allowed set if body types in NOTIFY requests, and to specify the default value to be used when there is no Accept header field in the SUBSCRIBE request. A NOTIFY for a call-completion package MUST contain a body that describes the call-completion states.

As described in [RFC3265], the NOTIFY message will contain bodies that describe the state of the subscribed resource. This body is in a format listed in the Accept header field of the SUBSCRIBE, or in a package-specific default format if the Accept header field was omitted from the SUBSCRIBE.

In this event package, the body of the notification contains a call-completion document. All subscribers and notifiers MUST support the "application/call-completion" data format described in section 8. The SUBSCRIBE request MAY contain an Accept header field. If no such header field is present, it has a default value of "application/call-completion". If the header field is present, it MUST include "application/call-completion". This "application/call-completion"
data format is described in chapter 8. Of course, the notifications generated by the server MUST be in one of the formats specified in the Accept header field in the SUBSCRIBE request.

6.6. Subscriber Generation of SUBSCRIBE Requests

Subscribers MUST generate SUBSCRIBE requests when they want to subscribe to the call-completion event package at the terminating side in order to receive call-completion notifications. The generation of SUBSCRIBE requests MAY imply the usage of call-completion service specific timers. An example of such an implementation can be found in ETSI TS 183 042.

6.7. Notifier Processing of SUBSCRIBE Requests

Upon receiving a subscription refresh, the notifier MUST set the "expires" parameter of the Subscription-State header to the current remaining duration of the subscription regardless of the value received in the Expires header (if present) of the subscription refresh.

If a subscription is not successful because the call-completion queue has reached the maximum number of entries (short term denial), the notifier MUST send a 480 Temporarily Unavailable response to the subscriber. If a subscription is not successful because a general error that prevents the call-completion service has occurred (long term denial), the notifier MUST send a 403 Forbidden response to the subscriber.

The call-completion information can be sensitive. Therefore, all subscriptions SHOULD be authenticated and then authorized before approval. The call-completion event package specified in this document is intended to be used in private domains (e.g. IMS) where authentication and authorization are provided via means out of scope of this document.

6.8. Notifier Generation of NOTIFY Requests

Notifiers MUST generate NOTIFY requests when a call-completion service condition occurs at the terminating side that needs to be sent towards the originating side.

A NOTIFY sent as a confirmation of the initial subscription or of a subscription refresh MUST contain the "call-completion-state" parameter set to "queued" if the user is busy and the call-completion subscription was successful (i.e. initial call-completion subscription, or a call-completion subscription for resume reasons) and to "ready-for-call-completion" if the call-completion target is
not busy.

A NOTIFY sent as a confirmation of a request to unsubscribe MAY contain the "call-completion-state" parameter.

When the callee’s status changes from busy to not busy, the notifier MUST send a NOTIFY only to first queue entry with an active subscription. This NOTIFY MUST contain the "call-completion-state" parameter set to "ready-for-call-completion".

If the call-completion subscription was successful and the retention option is supported at the callee, the NOTIFY MUST contain the "retention-option" parameter.

6.9. Subscriber Processing of NOTIFY Requests

The subscriber processing of NOTIFY requests MAY trigger additional CCBS service procedures (e.g. CCBS recall, usage of CCBS timers?). An example of such procedures can be found in ETSI TS 183 042.

6.10. Handling of Forked Requests

The SIP Events framework mandates that packages indicate whether or not forked SUBSCRIBE requests can install multiple subscriptions. Forked requests are NOT ALLOWED for the call completion event type.

6.11. Rate of Notifications

[RFC3265] mandates that packages define a maximum rate of notifications for their package. The call completion service typically involves a single notification per notifier and per subscription but MAY involve several notifications separated by a call completion call that failed due to a busy call completion target.

6.12. State Agents

[RFC3265] asks packages to consider the role of state agents in their design. State agents have no role in the handling of the call completion package.

7. Security Considerations

The use of the CC facility allows the caller’s agent to determine some status information regarding the callee. The information is confined to a busy/not-busy indication, and is to a considerable degree protected by the necessity of presenting the Call-Id of a
recent call to the callee in order to obtain information.

The CC facility may enhance the effectiveness of SPIT by the following technique: The caller makes calls to a group of targets. The caller then requests CC for the calls that do not connect to the targets. The CC calls resulting are probably more likely to reach the targets than original calls to a further group of targets.

8. IANA Considerations

This specification registers an event package, based on the registration procedures defined in . The followings is the information required for such a registration:

Package Name: call-completion

Package or Template-Package: This is a package.

Published Document: RFC XXXX(Note for RFC Editor: Please fill in XXXX with the RFC number of this specification).

Person to Contact: Martin Huelsemann, martin.huelsemann@t-com.net

9. References

9.1. Normative References

[I-D.ietf-sip-gruu] Rosenberg, J., "Obtaining and Using Globally Routable User
Agent (UA) URIs (GRUU) in the Session Initiation Protocol (SIP)",
draft-ietf-sip-gruu-15 (work in progress),
October 2007.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate

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


[RFC3515] Sparks, R., "The Session Initiation Protocol (SIP) Refer
9.2. Informative References


Authors' Addresses

Dale R. Worley
Pingtel Corp.
10 North Ave.
Burlington, MA 01803
US
Phone: +1 781 229 0533 x173
Email: dworley@pingtel.com
URI: http://www.pingtel.com

Martin Huelsemann
Deutsche Telekom
Deutsche-Telekom-Allee 1
Darmstadt 64307
Germany
Email: martin.huelsemann@t-com.net

Denis Alexeitsev
Deutsche Telekom
Deutsche-Telekom-Allee 1
Darmstadt 64307
Germany
Email: d.alexeitsev@t-com.net