Internet Engineering Task Force (IETF)
Request for Comments: 6228
Category: Standards Track
ISSN: 2070-1721

C. Holmberg
Ericsson

May 2011

Session Initiation Protocol (SIP) Response Code for Indication of Terminated Dialog

Abstract

This specification defines a new Session Initiation Protocol (SIP) response code, 199 Early Dialog Terminated, that a SIP forking proxy and a User Agent Server (UAS) can use to indicate to upstream SIP entities (including the User Agent Client (UAC)) that an early dialog has been terminated, before a final response is sent towards the SIP entities.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6228.

Table of Contents

1. Introduction

As defined in RFC 3261 [RFC3261], a Session Initiation Protocol (SIP) early dialog is created when a non-100 provisional response is sent to the initial dialog initiation request (e.g., INVITE, outside an existing dialog). The dialog is considered to be in early state until a final response is sent.

When a proxy receives an initial dialog initiation request, it can forward the request towards multiple remote destinations. When the proxy does that, it performs forking [RFC3261].

When a forking proxy receives a non-100 provisional response, or a 2xx final response, it forwards the response upstream towards the sender of the associated request. After a forking proxy has forwarded a 2xx final response, it normally generates and sends CANCEL requests downstream towards all remote destinations where it previously forked the request associated with the 2xx final response and from which it has still not received a final response. The CANCEL requests are sent in order to terminate any outstanding early dialogs associated with the request.

Upstream SIP entities might receive multiple 2xx final responses. When a SIP entity receives the first 2xx final response, and it does not intend to accept any subsequent 2xx final responses, it will automatically terminate any other outstanding early dialog associated with the request. If the SIP entity receives a subsequent 2xx final response, it will normally generate and send an ACK request, followed with a BYE request, using the dialog identifier retrieved from the 2xx final response.

> NOTE: A User Agent Client (UAC) can use the Request-Disposition header field [RFC3841] to request that proxies do not generate and send CANCEL requests downstream once they have received the first 2xx final response.

When a forking proxy receives a non-2xx final response, it does not always immediately forward the response upstream towards the sender of the associated request. Instead, the proxy "stores" the response and waits for subsequent final responses from other remote destinations where the associated request was forked. At some point, the proxy uses a specified mechanism to determine the "best" final response code, and forwards a final response using that response code upstream towards the sender of the associated request. When an upstream SIP entity receives the non-2xx final response, it will release resources associated with the session. The UAC will terminate, or retry, the session setup.

Since the forking proxy does not always immediately forward non-2xx final responses, upstream SIP entities (including the UAC that initiated the request) are not immediately informed that an early dialog has been terminated, and will therefore maintain resources associated with the early dialog reserved until a final response is sent by the proxy, even if the early dialog has already been terminated. A SIP entity could use the resources for other things, e.g., to accept subsequent early dialogs that it otherwise would reject.

This specification defines a new SIP response code, 199 Early Dialog Terminated. A forking proxy can send a 199 provisional response to inform upstream SIP entities that an early dialog has been terminated. A UAS can send a 199 response code, prior to sending a non-2xx final response, for the same purpose. SIP entities that receive the 199 response can use it to trigger the release of resources associated with the terminated early dialog. In addition, SIP entities might also use the 199 response to make policy decisions related to early dialogs. For example, a media gate controlling a SIP entity might use the 199 response when deciding for which early dialogs media will be passed.

Section 9 contains signalling examples that show when and how a forking proxy generates 199 responses in different situations.

2. 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 RFC 2119 [RFC2119].

3. Applicability and Limitation

The 199 response code is an optimization, and it only optimizes how quickly recipients might be informed about terminated early dialogs. The achieved optimization is limited. Since the response is normally not sent reliably by a UAS, and cannot be sent reliably when generated and sent by a proxy, it is possible that some or all of the 199 responses will get lost before they reach the recipients. In such cases, recipients will behave the same as if the 199 response code were not used at all.

One example for which a UAC could use the 199 response is that when it receives a 199 response, it releases resources associated with the terminated early dialog. The UAC could also use the 199 response to make policy decisions related to early dialogs. For example, if a UAC is playing media associated with an early dialog, and it then receives a 199 response indicating the early dialog has been terminated, it could start playing media associated with a different early dialog.

Application designers utilizing the 199 response code MUST ensure that the application’s user experience is acceptable if all 199 responses are lost and not delivered to the recipients.

4. User Agent Client Behavior

When a UAC sends an initial dialog initiation request, and if it is willing to receive 199 responses, it MUST insert a "199" option-tag in the Supported header field [RFC3261] of the request. The option-tag indicates that the UAC supports, and is willing to receive, 199 responses. A UAC SHOULD NOT insert a "199" option-tag in the Require or the Proxy-Require header field [RFC3261] of the request, since in many cases it would result in unnecessary session establishment failures.

> NOTE: The UAC always needs to insert a "199" option-tag in the Supported header field, in order to indicate that it supports, and is willing to receive, 199 responses, even if it also inserts the option-tag in the Require or Proxy-Require header field.

It is RECOMMENDED that a UAC not insert a "100rel" option-tag [RFC3262] in the Require header field when it also indicates support for 199 responses, unless the UAC also uses some other SIP extension or procedure that mandates it to do so. The reason is that proxies are not allowed to generate and send 199 responses when the UAC has required provisional responses to be sent reliably.

When a UAC receives a 199 response, it might release resources associated with the terminated early dialog. A UAC might also use the 199 response to make policy decisions related to early dialogs.

NOTE: The 199 response indicates that the early dialog has been terminated, so there is no need for the UAC to send a BYE request in order to terminate the early dialog when it receives the 199 response.

NOTE: The 199 response does not affect other early dialogs associated with the session establishment. For those dialogs, the normal SIP rules regarding transaction timeout, etc., still apply.

Once a UAC has received and accepted a 199 response, it MUST NOT send any media associated with the early dialog. In addition, if the UAC is able to associate received media with early dialogs, it MUST NOT process any received media associated with the early dialog that was terminated.

If multiple usages [RFC5057] are used within an early dialog, and it is not clear which dialog usage the 199 response terminates, SIP entities that keep dialog state SHALL NOT release resources associated with the early dialog when they receive the 199 response.

If a UAC receives an unreliably sent 199 response on a dialog that has not previously been established (this can happen if a 199 response reaches the client before the 18x response that would establish the early dialog) it SHALL discard the 199 response. If a UAC receives a reliably sent 199 response on a dialog that has not previously been created, it MUST acknowledge the 199 response, as described in RFC 3262 [RFC3262].

If a UAC has received a 199 response for all early dialogs, and no early dialogs associated with the session establishment remain, it maintains the "Proceeding" state [RFC3261] and waits for possible subsequent early dialogs to be established, and eventually for a final response to be received.

5. User Agent Server Behavior

If a UAS receives an initial dialog initiation request with a Supported header field that contains a "199" option-tag, it SHOULD NOT send a 199 response on an early dialog associated with the request before it sends a non-2xx final response. Cases where a UAS might send a 199 response are if it has been configured to do so due to lack of support for the 199 response code by forking proxies or other intermediate SIP entities, or if it is used in an environment that specifies that it shall send a 199 response before sending a non-2xx response.

> NOTE: If a UAS has created multiple early dialogs associated with an initial dialog initiation request (the UAS is acting similarly to a forking proxy), it does not always intend to send a final response on all of those early dialogs.
>
> NOTE: If the Require header field of an initial dialog initiation request contains a "100rel" option-tag, proxies will not be able to generate and send 199 responses. In such cases, the UAS might choose to send a 199 response on an early dialog before it sends a non-2xx final response, even if it would not do so in other cases.

If the Supported header field of an initial dialog initiation request does not contain a "199" option-tag, the UAC MUST NOT send a 199 response on any early dialog associated with the request.

When a UAS generates a 199 response, the response MUST contain a To header field tag parameter [RFC3261], in order for other entities to identify the early dialog that has been terminated. The UAS MUST also insert a Reason header field [RFC3326] that contains a response code describing the reason why the early dialog was terminated. The UAS MUST NOT insert a "199" option-tag in the Supported, Require, or Proxy-Require header field of the 199 response.

If a UAS intends to send 199 responses, and if it supports the procedures defined in RFC 3840 [RFC3840], it MAY during the registration procedure use the sip.extensions feature tag [RFC3840] to indicate support for the 199 response code.

A 199 response SHOULD NOT contain a Session Description Protocol (SDP) offer/answer message body, unless required by the rules in RFC 3264 [RFC3264].

According to RFC 3264, if an INVITE request does not contain an SDP offer, and the 199 response is the a first reliably sent response associated with the request, the 199 response is required to contain an SDP offer. In this case, the UAS SHOULD send the 199 response unreliably, or send the 199 response reliably and include an SDP offer with no "m=" lines in the response.

Since a 199 response is only used for information purposes, the UAS SHOULD send it unreliably, unless the "100rel" option-tag is present in the Require header field of the associated request.

6. Proxy Behavior

When a proxy receives a 199 response to an initial dialog initiation request, it MUST process the response as any other non-100 provisional response. The proxy will forward the response upstream towards the sender of the associated request. The proxy MAY release resources it has reserved associated with the early dialog that is terminated. If a proxy receives a 199 response out of dialog, it MUST process it as other non-100 provisional responses received out of dialog.

When a forking proxy receives a non-2xx final response to an initial dialog initiation request that it recognizes as terminating one or more early dialogs associated with the request, it MUST generate and send a 199 response upstream for each of the terminated early dialogs that satisfy each of the following conditions:

- The forking proxy does not intend to forward the final response immediately (in accordance with rules for a forking proxy).

- The UAC has indicated support (by inserting the "199" option-tag in a Supported header field) for the 199 response code in the associated request.

- The UAC has not required provisional responses to be sent reliably (i.e., has not inserted the "100rel" option-tag in a Require or Proxy-Require header field) in the associated request.

- The forking proxy has not already received and forwarded a 199 response for the early dialog.

- The forking proxy has not already sent a final response for any of the early dialogs.

As a consequence, once a final response to an initial dialog initiation request has been issued by the proxy, no further 199 responses associated with the request will be generated or forwarded by the proxy.

When a forking proxy forks an initial dialog initiation request, it generates a unique Via header branch parameter value for each forked leg. A proxy can determine whether additional forking has occurred downstream of the proxy by storing the top Via branch value from each response that creates an early dialog. If the same top Via branch value is received for multiple early dialogs, the proxy knows that additional forking has occurred downstream of the proxy. A non-2xx final response received for a specific early dialog also terminates all other early dialogs for which the same top Via branch value was received in the responses that created those early dialogs.

Based on implementation policy, a forking proxy MAY wait before sending the 199 response, e.g., if it expects to receive a 2xx final response on another dialog shortly after it received the non-2xx final response that triggered the 199 response.

When a forking proxy generates a 199 response, the response MUST contain a To header field tag parameter that identifies the terminated early dialog. A proxy MUST also insert a Reason header field that contains the SIP response code of the response that triggered the 199 response. The SIP response code in the Reason header field informs the receiver of the 199 response about the SIP response code that was used by the UAS to terminate the early dialog, and the receiver might use that information for triggering different types of actions and procedures. The proxy MUST NOT insert a "199" option-tag in the Supported, Require, or Proxy-Require header field of the 199 response.

A forking proxy that supports the generation of 199 responses MUST keep track of early dialogs, in order to determine whether to generate a 199 response when the proxy receives a non-2xx final response. In addition, a proxy MUST keep track on which early dialogs it has received and forwarded 199 responses, in order to not generate additional 199 responses for those early dialogs.

If a forking proxy receives a reliably sent 199 response for a dialog for which it has previously generated and sent a 199 response, it MUST forward the 199 response. If a proxy receives an unreliably sent 199 response for which it has previously generated and sent a 199 response, it MAY forward the response, or it MAY discard it.

When a forking proxy generates and sends a 199 response, the response SHOULD NOT contain a Contact header field or a Record-Route header field [RFC3261].

If the Require header field of an initial dialog initiation request contains a "100rel" option-tag, a proxy MUST NOT generate and send 199 responses associated with that request. The reason is that a proxy is not allowed to generate and send 199 responses reliably.

7. Backward Compatibility

Since all SIP entities involved in a session setup do not necessarily support the specific meaning of the 199 Early Dialog Terminated provisional response, the sender of the response MUST be prepared to receive SIP requests and responses associated with the dialog for which the 199 response was sent (a proxy can receive SIP messages from either direction). If such a request is received by a UA, it MUST act in the same way as if it had received the request after sending the final non-2xx response to the INVITE request, as specified in RFC 3261. A UAC that receives a 199 response for an early dialog MUST NOT send any further requests on that dialog, except for requests that acknowledge reliable responses. A proxy MUST forward requests according to RFC 3261, even if the proxy has knowledge that the early dialog has been terminated.

A 199 response does not "replace" a final response. RFC 3261 specifies when a final response is sent.

8. Usage with SDP Offer/Answer

A 199 response SHOULD NOT contain an SDP offer/answer [RFC3264] message body, unless required by the rules in RFC 3264.

If an INVITE request does not contain an SDP offer, and the 199 response is the first reliably sent response, the 199 response is required to contain an SDP offer. In this case, the UAS SHOULD send the 199 response unreliably, or include an SDP offer with no "m=" lines in a reliable 199 response.

9. Message Flow Examples

9.1. Example with a Forking Proxy that Generates 199

Figure 1 shows an example where a proxy (P1) forks an INVITE received from a UAC. The forked INVITE reaches UAS_2, UAS_3, and UAS_4, which send 18x provisional responses in order to establish early dialogs between themselves and the UAC. UAS_2 and UAS_3 each reject the INVITE by sending a 4xx error response. When P1 receives the 4xx

responses, it immediately sends 199 responses towards the UAC, to indicate that the early dialogs for which it received the 4xx responses have been terminated. The early dialog leg is shown in parentheses.

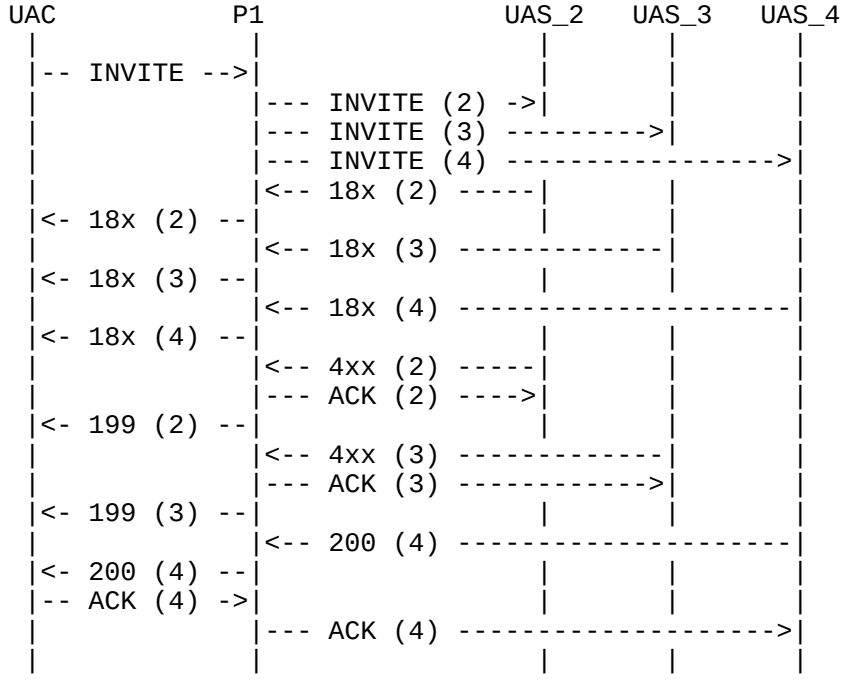


Figure 1: Example Call Flow

9.2. Example with a Forking Proxy that Receives 200 OK

Figure 2 shows an example where a proxy (P1) forks an INVITE request received from a UAC. The forked request reaches UAS_2, UAS_3, and UAS_4, all of which send 18x provisional responses in order to establish early dialogs between themselves and the UAC. Later, UAS_4 accepts the session and sends a 200 OK final response. When P1 receives the 200 OK response, it immediately forwards it towards the UAC. P1 does not send 199 responses for the early dialogs from UAS_2 and UAS_3, since P1 has still not received any final responses on those early dialogs (even if P1 sends CANCEL requests to UAS_2 and UAS_3, P1 may still receive a 200 OK final response from UAS_2 or UAS_3, which P1 would have to forward towards the UAC. The early dialog leg is shown in parentheses.

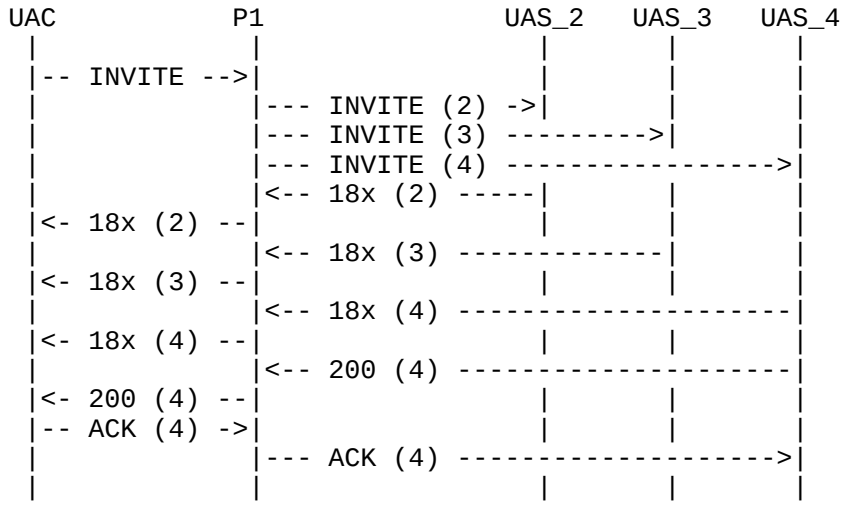
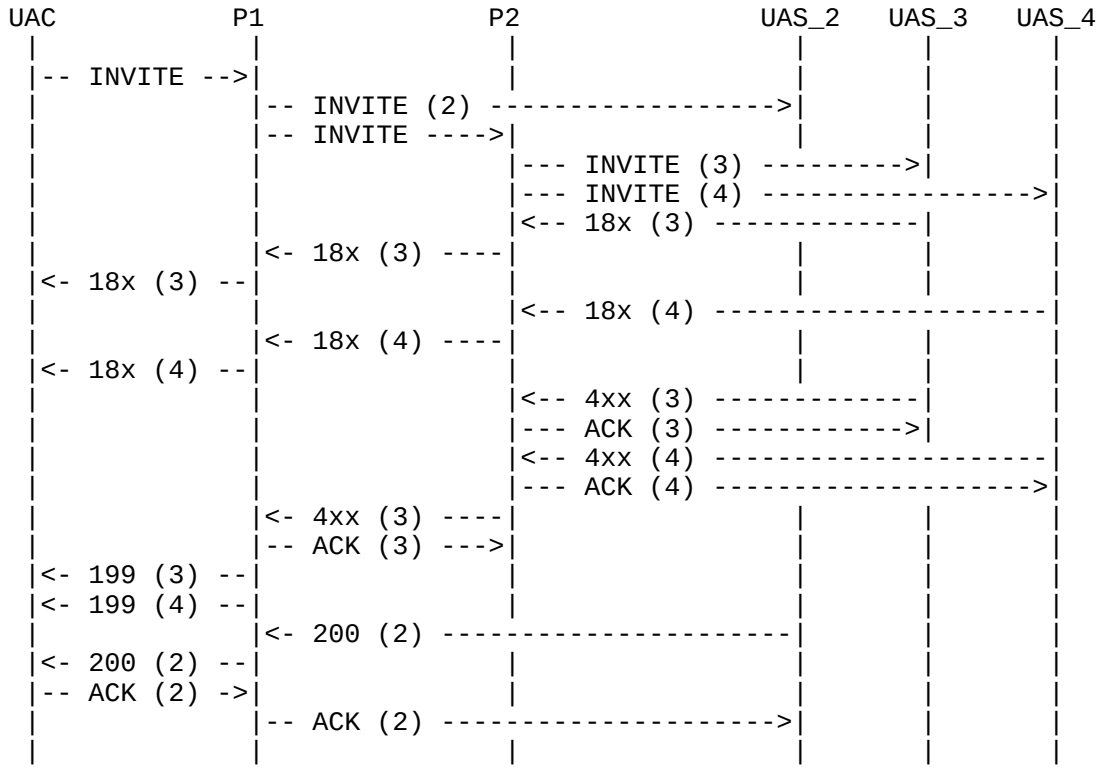


Figure 2: Example Call Flow

9.3. Example with Two Forking Proxies, of which One Generates 199

Figure 3 shows an example where a proxy (P1) forks an INVITE request received from a UAC. One of the forked requests reaches UAS_2. The other requests reach another proxy (P2), which forks the request to UAS_3 and UAS_4. UAS_3 and UAS_4 send 18x provisional responses in order to establish early dialogs between themselves and the UAC. Later, UAS_3 and UAS_4 each reject the INVITE request by sending a 4xx error response. P2 does not support the 199 response code and forwards a single 4xx response. P1 supports the 199 response code, and when it receives the 4xx response from P2, it also manages to associate the early dialogs from both UAS_3 and UAS_4 with the response. Therefore, P1 generates and sends two 199 responses to indicate that the early dialogs from UAS_3 and UAS_4 have been terminated. The early dialog leg is shown in parentheses.



```
    UAC           P1              P2                UAS_2   UAS_3   UAS_4
     |             |               |                 |       |       |
     |-- INVITE -->|               |                 |       |       |
     |             |-- INVITE (2) ------------------>|       |       |
     |             |-- INVITE ---->|                 |       |       |
     |             |               |--- INVITE (3) --------->|       |
     |             |               |--- INVITE (4) ----------------->|
     |             |               |<-- 18x (3) -------------|       |
     |             |<- 18x (3) ----|                 |       |       |
     |<- 18x (3) --|               |                 |       |       |
     |             |               |<-- 18x (4) ---------------------|
     |             |<- 18x (4) ----|                 |       |       |
     |<- 18x (4) --|               |                 |       |       |
     |             |               |<-- 4xx (3) -------------|       |
     |             |               |--- ACK (3) ------------>|       |
     |             |               |<-- 4xx (4) ---------------------|
     |             |               |--- ACK (4) -------------------->|
     |             |<- 4xx (3) ----|                 |       |       |
     |             |-- ACK (3) --->|                 |       |       |
     |<- 199 (3) --|               |                 |       |       |
     |<- 199 (4) --|               |                 |       |       |
     |             |<- 200 (2) ----------------------|       |       |
     |<- 200 (2) --|               |                 |       |       |
     |-- ACK (2) ->|               |                 |       |       |
     |             |-- ACK (2) --------------------->|       |       |
     |             |               |                 |       |       |
```

Figure 3: Example Call Flow

10. Security Considerations

General security issues related to SIP responses are described in RFC 3261. Due to the nature of the 199 response, it may be attractive to use it for launching attacks in order to terminate specific early dialogs (other early dialogs will not be affected). In addition, if a man-in-the-middle generates and sends towards the UAC a 199 response that terminates a specific dialog, it can take a while until the UAS finds out that the UAC, and possible stateful intermediates, have terminated the dialog. SIP security mechanisms (e.g., hop-to-hop Transport Layer Security (TLS)) can be used to minimize, or eliminate, the risk of such attacks.

11. IANA Considerations

This section registers a new SIP response code and a new option-tag, according to the procedures of RFC 3261.

11.1. IANA Registration of the 199 Response Code

This section registers a new SIP response code, 199. The required information for this registration, as specified in RFC 3261, is:

RFC Number: RFC 6228

Response Code Number: 199

Default Reason Phrase: Early Dialog Terminated

11.2. IANA Registration of the 199 Option-Tag

This section registers a new SIP option-tag, 199. The required information for this registration, as specified in RFC 3261, is:

Name: 199

Description: This option-tag is for indicating support of the 199 Early Dialog Terminated provisional response code. When present in a Supported header of a request, it indicates that the UAC supports the 199 response code. When present in a Require or Proxy-Require header field of a request, it indicates that the UAS, or proxies, MUST support the 199 response code. It does not require the UAS, or proxies, to actually send 199 responses.

12. Acknowledgements

Thanks to Paul Kyzivat, Dale Worley, Gilad Shaham, Francois Audet, Attila Sipos, Robert Sparks, Brett Tate, Ian Elz, Hadriel Kaplan, Timothy Dwight, Dean Willis, Serhad Doken, John Elwell, Gonzalo Camarillo, Adam Roach, Bob Penfield, Tom Taylor, Ya Ching Tan, Keith Drage, Hans Erik van Elburg, and Cullen Jennings for their feedback and suggestions.

13. References

13.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[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.

[RFC3262] Rosenberg, J. and H. Schulzrinne, "Reliability of Provisional Responses in Session Initiation Protocol (SIP)", RFC 3262, June 2002.

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, June 2002.

[RFC3326] Schulzrinne, H., Oran, D., and G. Camarillo, "The Reason Header Field for the Session Initiation Protocol (SIP)", RFC 3326, December 2002.

[RFC3840] Rosenberg, J., Schulzrinne, H., and P. Kyzivat, "Indicating User Agent Capabilities in the Session Initiation Protocol (SIP)", RFC 3840, August 2004.

13.2. Informative References

[RFC3841] Rosenberg, J., Schulzrinne, H., and P. Kyzivat, "Caller Preferences for the Session Initiation Protocol (SIP)", RFC 3841, August 2004.

[RFC5057] Sparks, R., "Multiple Dialog Usages in the Session Initiation Protocol", RFC 5057, November 2007.

Author’s Address

Christer Holmberg
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

EMail: christer.holmberg@ericsson.com