DELETING MECHANISM IN SIP MULTIMEDIA SERVICES

FIELD OF THE INVENTION

[0001] The present invention relates generally to session initiation protocol (SIP) services and SIP for instant messaging and presence leveraging extensions (SIMPLE) services. More particularly, the present invention relates to SIP/SIMPLE based services such as instant messaging (IM) and push-to-talk (PoC) services.

BACKGROUND OF THE INVENTION

[0002] This section is intended to provide a background or context to the invention that is recited in the claims. The description herein may include concepts that could be pursued, but are not necessarily ones that have been previously conceived or pursued. Therefore, unless otherwise indicated herein, what is described in this section is not prior art to the description and claims in this application and is not admitted to be prior art by inclusion in this section.

[0003] The Open Mobile Alliance (OMA) is a standards body which collectively develops open standards for use in the mobile industry. The OMA helps to create interoperable services enablers to work across countries, operators and mobile terminals and is driven by market requirements. To expand the mobile market, companies supporting the Open Mobile Alliance work to aid in the rapid and wide development and deployment of a variety of new, enhanced mobile information, communication and entertainment services.

[0004] The OMA is currently developing IM services based on SIP, Message Session Relay Protocol (MSRP) and Extensible Markup Language (XML) Configuration Access Protocol (XCAP) protocols developed by the International Engineering Task Force (IETF) SIMPLE working group. Instant Messaging service are already deployed using several proprietary technologies and Wireless Village specifications.

[0005] There is currently a need for a deleting Mechanism in the SIP multimedia service environment. In an http environment, if a document needs to be deleted, an "http delete" command is simply issued. However, there is currently no corresponding deleting feature or function defined for the SIP environment. In fact, even SIP extensions for services have not defined such a feature. In current multimedia services, particularly OMA SIP/SIMPLE IM, there are several requirements about storing and retrieving messages. Although there is a need for deleting and selectively deleting stored messages, such a mechanism has yet to be defined.

SUMMARY OF THE INVENTION

[0006] The present invention comprises a novel deleting mechanism for use in SIP multimedia services. The present invention involves the use of various SIP multimedia service environment features for this purpose. In one embodiment, a "recycle bin" is defined in the network and is associated with a SIP uniform resource identifier. Messages that are stored within the network are assigned a unique identifier. If a user desires to delete the message, he or she requests that a SIP/MSRP function be set up between the message and the network-defined recycle bin. When processed, the message is transferred to the recycle bin, leaving the user's account in the user's mail storage server.

[0007] The system and method of the present invention is simple and easy to adopt, as already-existing defined tools such as the SIP REFER method, Virtual User Agent and SIP URI are used.

[0008] These and other advantages and features of the invention, together with the organization and manner of operation thereof, will become apparent from the following detailed description when taken in conjunction with the accompanying drawings, wherein like elements have like numerals throughout the several drawings described below.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] Figure 1 is a flow chart showing the operation of a deleting mechanism for SIP multimedia services in accordance with one embodiment of the present invention;

[0010] Figure 2 is a flow chart showing the operation of a deleting mechanism for SIP multimedia services for deleting a selected message in accordance with another embodiment of the present invention;

[0011] Figure 3 is a flow chart showing the operation of a deleting mechanism for SIP multimedia services for deleting a selected message in accordance with still another embodiment of the present invention;

[0012] Figure 4 is a flow chart showing the operation of a deleting mechanism for SIP multimedia services for deleting multiple selected messages in accordance with an embodiment of the present invention;

[0013] Figure 5 is a flow chart showing the operation of a deleting mechanism for SIP multimedia services for deleting multiple selected messages in accordance with an embodiment of the present invention;

[0014] Figure 6 is a flow chart showing the operation of a deleting mechanism for SIP multimedia services for deleting multiple selected messages in accordance with an embodiment of the present invention;

[0015] Figure 7 is a flow chart showing the operation of a deleting mechanism for SIP multimedia services for deleting all messages in a user's mail storage account in accordance with an embodiment of the present invention;

[0016] Figure 8 is an overview diagram of a system within which the present invention may be implemented;

[0017] Figure 9 is a perspective view of a mobile telephone that can be used in the implementation of the present invention;

[0018] Figure 10 is a schematic representation of the telephone circuitry of the mobile telephone of Figure 9;

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0019] The present invention comprises a novel deleting mechanism for use in SIP multimedia services. The present invention involves the use of various SIP

multimedia service environment features for this purpose. In one embodiment, a "recycle bin" or similar location for deleted items is defined in the network and is associated with a SIP uniform resource identifier. Messages that are stored within the network are assigned a unique identifier. If a user desires to delete the message, he or she requests that a SIP/MSRP session function be set up between the message and the network-defined recycle bin. When processed, the message is transferred to the recycle bin, leaving the user's account in the user's mail storage server.

[0020] Figure 1 is a flow chart showing the operation of a deleting mechanism for SIP multimedia services in accordance with one embodiment of the present invention. In particular, Figure 1 shows the interaction among a user/client device 100, a user's account in mail storage 110 and a recycle bin 120 as defined herein. Both the user's account 110 and the recycle bin 120 are located remote from the user/client device. In the embodiment shown in Figure 1, the SIP URI for the user/client device is "User@Sonera.com." The SIP URI for the user's account is "User@mailserver57.Sonera.com." The SIP URI for the recycle bin is "RecycleBin@mailserver.sonera.com."

[0021] As discussed above, stored messages in the network can be assigned unique message identifiers. Three such messages are shown at 130 in Figure 1 with identifiers of "13242@mailserver57.Sonera.com" (MSG 1) "13243@mailserver57.Sonera.com" (MSG 2) and "13244@mailserver57.Sonera.com" (MSG 3). Alternatively, messages can be stored as files. In one embodiment, each message can be given a file name, a file type, and a hash value. Three such messages are shown in Figure 2 with identifiers of "File 1 = (filename, filetype, unique hash value)", "File 2 = (filename, filetype, unique hash value)", and "File 3 = (filename, filetype, unique hash value)".

[0022] At 140 in Figure 1, a user decides he or she wants to delete MSG 2. At this point, the user/client device 100 sends an SIP REFER with INVITE request 150 to the message identifier 13243@mailserver57.Sonera.com, which serves as a virtual user agent 155, at the user's account 110. The SIP REFER request has the network-based recycle bin address (RecycleBin@mailserver.sonera.com) in the Refer-to header. The SIP REFER with INVITE request 150 serves to request that a SIP Session be set up

with the network-based recycle bin 120 (RecycleBin@mailserver.sonera.com). The virtual user agent 155 responds by accepting the SIP REFER request from the user/client device 100 with a "202 ACCEPT" message at 160. The virtual user agent 155 also sends an INVITE request to set up a SIP session with the recycle bin 120 at 170. The recycle bin 120 accepts this session at 180. At 190, an SIP session is officially set up with the virtual user agent 155 in the form of a message session relay protocol (MSRP), with the session description protocol (SDP) media attribute set to a=SendOnly. The virtual user agent 155 proceeds to notify the user/client 100 of the SIP session at 200, and the user/client device 100 acknowledges this notification at 210. In the SIP/MSRP session, MSG 2 is sent from the user's account 110 to the network-based recycle bin 120, causing MSG 2 to disappear from the user's account 110. After successful transmission of the message MSG2, the SIP session between the virtual user agent 155 and the recycle bin 120 is torn down. The end result, depicted at 220, is the presence of only MSG 1 and MSG 3 in the user's account 110 in the user's mail storage server.

[0023] In an alternative embodiment of the present invention, the functions of the user's account 110 and the recycle bin 120 are collocated. In this situation, the sending of an INVITE request to set up a SIP session 170, the acknowledgement of this request 180, and the setting up of the SIP session with MSRP 190 is not necessary.

[0024] An alternative embodiment in which the messages are stored as files is illustrated in Figure 2. In this embodiment the mechanism to retrieve the stored or selected messages to the recycle bin can be based on the file transfer draft attached as Exhibit B, which is incorporated into this application. In this embodiment, the REFER request can include the SDP descriptions of the file(s) to be deleted.

[0025] At 140 in Figure 2, a user decides he or she wants to delete MSG 2 (File 2). At this point, the user/client device 100 sends an SIP REFER with INVITE request 150 to the mail storage server, which serves as a virtual user agent 155, at the user's account 110. The SIP REFER request has the network-based recycle bin address (RecycleBin@mailserver.sonera.com) in the Refer-to header. The SIP REFER with INVITE request 150 serves to request that a SIP Session be set up with the network-

based recycle bin 120 (RecycleBin@mailserver.sonera.com). The virtual user agent 155 responds by accepting the SIP REFER request from the user/client device 100 with a "202 ACCEPT" message at 160. The virtual user agent 155 also sends an INVITE request to set up a SIP session with the recycle bin 120 at 170. The recycle bin 120 accepts this session at 180. At 190, an SIP session is officially set up with the virtual user agent 155 in the form of a message session relay protocol (MSRP), with the session description protocol (SDP) media attribute set to a=SendOnly. The virtual user agent 155 proceeds to notify the user/client 100 of the SIP session at 200, and the user/client device 100 acknowledges this notification at 210. In the SIP/MSRP session, File 2 is sent from the user's account 110 to the network-based recycle bin 120, causing MSG 2 to disappear from the user's account 110. After successful transmission of the file File 2 (MSG2), the SIP session between the virtual user agent 155 and the recycle bin 120 is torn down. The end result, depicted at 220, is the presence of only MSG 1 (File 1) and MSG 3 (File 3) in the user's account 110 in the user's mail storage server.

[0026] In an alternative embodiment of the present invention, the functions of the user's account 110 and the recycle bin 120 are collocated. In this situation, the sending of an INVITE request to set up a SIP session 170, the acknowledgement of this request 180, and the setting up of the SIP session with MSRP 190 is not necessary.

[0027] An alternative embodiment of the invention is illustrated in Figure 3. In this embodiment the SIP REFER with INVITE request is sent directly to the network-based recycle bin bypassing the virtual user agent. For example, at 340 in Figure 3, a user decides he or she wants to delete MSG 2. At this point, the user/client device 100 sends an SIP REFER with INVITE request 350 to the recycle bin 120 (RecycleBin@mailserver.sonera.com). The SIP REFER with INVITE request 350 serves to request that a SIP Session be set up between the network-based recycle bin 120 (RecycleBin@mailserver.sonera.com) and the user's account 110 or the virtual user agent 155 if one is used. The recycle bin 120 responds by accepting the SIP REFER request from the user/client device 100 with a "202 ACCEPT" message at 360. The recycle bin 120 also sends an INVITE request to set up a SIP session

with virtual user agent 155 at 370. The virtual user agent 155 accepts this session at 380. At 390, an SIP session is officially set up with the virtual user agent 155 in the form of a message session relay protocol (MSRP), with the session description protocol (SDP) media attribute set to a=RecvOnly. The recycle bin 120 proceeds to notify the user/client 100 of the SIP session at 400, and the user/client device 100 acknowledges this notification at 410. In the SIP/MSRP session, MSG 2 is sent from the user's account 110 to the network-based recycle bin 120, causing MSG 2 to disappear from the user's account 110. After successful transmission of the message MSG2, the SIP session between the virtual user agent 155 and the recycle bin 120 is torn down. The end result, depicted at 420, is the presence of only MSG 1 and MSG 3 in the user's account 110 in the user's mail storage server.

[0028] Similar to the earlier embodiment, alternatively the functions of the user's account 110 and the recycle bin 120 are collocated. In this situation, the sending of an INVITE request to set up a SIP session 370, the acknowledgement of this request 380, and the setting up of the SIP session with MSRP 390 is not necessary.

[0029] The embodiment of the invention illustrated in Figure 4 is an alternative to the embodiment of Figure 3. Similar to Figure 3, in this embodiment the SIP REFER with INVITE request is sent directly to the network-based recycle bin bypassing the virtual user agent. However, in the embodiment in Figure 4, the mechanism to retrieve the stored or selected messages to the recycle bin is based on the file transfer draft attached as Exhibit B. In this embodiment, the REFER request includes the SDP descriptions of the file(s) to be deleted.

[0030] At 340 in Figure 4, a user decides he or she wants to delete MSG 2 (File 2). At this point, the user/client device 100 sends an SIP REFER with INVITE request 350 to the recycle bin 120 (RecycleBin@mailserver.sonera.com). The SIP REFER with INVITE request 350 serves to request that a SIP Session be set up with the network-based recycle bin 120 (RecycleBin@mailserver.sonera.com). The recycle bin 120 responds by accepting the SIP REFER request from the user/client device 100 with a "202 ACCEPT" message at 360. The recycle bin 120 also sends an INVITE request to set up a SIP session with virtual user agent 155 at 370. The virtual user agent 155 accepts this session at 380. At 390, an SIP session is officially set up with

the virtual user agent 155 in the form of a message session relay protocol (MSRP), with the session description protocol (SDP) media attribute set to a=RecOnly. The recycle bin 120 proceeds to notify the user/client 100 of the SIP session at 400, and the user/client device 100 acknowledges this notification at 410. In the SIP/MSRP session, File 2 (MSG 2) is sent from the user's account 110 to the network-based recycle bin 120, causing File 2 (MSG 2) to disappear from the user's account 110. After successful transmission of File 2 (MSG2), the SIP session between the virtual user agent 155 and the recycle bin 120 is torn down. The end result, depicted at 420, is the presence of only File 1 (MSG 1) and File 3 (MSG 3) in the user's account 110 in the user's mail storage server.

[0031] Similar to the earlier embodiment, alternatively the functions of the user's account 110 and the recycle bin 120 are collocated. In this situation, the sending of an INVITE request to set up a SIP session 370, the acknowledgement of this request 380, and the setting up of the SIP session with MSRP 390 is not necessary.

[0032] In another embodiment, multiple stored messages can be selected and deleted by the user. In this embodiment, illustrated in Figure 5 a Multiple-REFER request can be sent to the recycle bin 120 to delete multiple selected messages.. Exhibit A, which is incorporated into this application, illustrates one embodiment or implementation of the Multiple-REFER request. In the embodiment shown in Figure 5, the SIP Multiple-REFER with INVITE request is sent directly to the network-based recycle bin bypassing the virtual user agent. Alternatively, the SIP Multiple-REFER with INVITE request can be sent to the virtual user agent as described with respect to the embodiment illustrated in Figure 1.

[0033] At 540 in Figure 5, a user decides he or she wants to delete MSG 2 and MSG 3. At this point, the user/client device 100 sends an SIP Multiple-REFER with INVITE request 550 to the recycle bin 120 (RecycleBin@mailserver.sonera.com) including a URI list containing the URIs of the stored messages to be deleted (in this case MSG 2 and MSG3). The SIP Multiple-REFER with INVITE request 550 serves to request that SIP Sessions be set up with the network-based recycle bin 120 (RecycleBin@mailserver.sonera.com). The recycle bin 120 sends INVITE requests to set up SIP sessions with virtual user agents 155 and 156 at 570 and 571,

respectively, one for each message being deleted. In this case, INVITE request 570 corresponds to MSG2 and INVITE request 571 corresponds to MSG3. The virtual user agents 155 and 156 accept these sessions at 580 and 581, respectively. At 590, a SIP session is officially set up with the virtual user agent 155 in the form of a message session relay protocol (MSRP), with the session description protocol (SDP) media attribute set to a=RecvOnly and at 591 a SIP session is set up with virtual user agent 156. In the SIP/MSRP sessions, MSG 2 and MSG3 are sent from the user's account 110 to the network-based recycle bin 120, causing MSG 2 and MSG3 to disappear from the user's account 110. After successful transmission of the messages MSG2 and MSG3, the SIP session between the virtual user agents 155 and 156 and the recycle bin 120 are torn down. The end result, depicted at 620, is the presence of only MSG 1 in the user's account 110 in the user's mail storage server.

[0034] Similar to the earlier embodiments, the functions of the user's account 110 and the recycle bin 120 can also be collocated. In this situation, the sending of the INVITE requests to set up a SIP session 570 and 571, the acknowledgements of these requests 580 and 581, and the setting up of the SIP sessions with MSRP 590 and 591 are not necessary.

[0035] The embodiment shown in Figure 6 illustrates deleting multiple messages when the mechanism to retrieve the stored or selected messages to the recycle bin is based on the file transfer draft attached as Exhibit B. In this embodiment, the REFER request again includes the SDP descriptions of the files to be deleted. At 540 in Figure 6, a user decides he or she wants to delete MSG 2 (File 2) and MSG 3 (File 3). At this point, the user/client device 100 sends an SIP REFER with INVITE request 550 to the recycle bin 120 (RecycleBin@mailserver.sonera.com) using the syntax described in Exhibit B for the stored messages (files) to be deleted (in this case MSG 2 and MSG3). In this case, the SDP parameters for each file (message) that is to be deleted needs to be sent in a separate media line "m =". The SIP REFER with INVITE request 550 serves to request that SIP Sessions be set up between the network-based recycle bin 120 (RecycleBin@mailserver.sonera.com) and the user's account 110 or the virtual user agent 155 if one is used. The recycle bin 120 sends INVITE requests to set up SIP sessions with virtual user agent 155 at 570. The

virtual user agent 155 accepts a session for each file to be deleted at 580 and 581, respectively. At 590, a SIP session is officially set up with the virtual user agent 155 in the form of a message session relay protocol (MSRP), with the session description protocol (SDP) media attribute set to a=RecvOnly and at 591 a SIP session is set up with virtual user agent 156. In the SIP/MSRP sessions, File 2 (MSG 2) and File 3 (MSG3) are sent from the user's account 110 to the network-based recycle bin 120, causing MSG 2 (File 2) and MSG3 (File 3) to disappear from the user's account 110. After successful transmission of the files File 2 (MSG 2) and File 3 (MSG3), the SIP sessions between the virtual user agent 155 and the recycle bin 120 are torn down. The end result, depicted at 620, is the presence of only File 1 (MSG 1) in the user's account 110 in the user's mail storage server.

[0036] Similar to the earlier embodiments, the functions of the user's account 110 and the recycle bin 120 can also be collocated. In this situation, the sending of the INVITE requests to set up a SIP session 570 and 571, the acknowledgements of these requests 580 and 581, and the setting up of the SIP sessions with MSRP 590 and 591 are not necessary.

[0037] In still another embodiment, all stored messages can be selected and deleted by the user. In this embodiment, illustrated in Figure 7 a REFER request can be sent to the recycle bin 120 to delete all messages by referring the SIP URI for the user's mail storage account instead of an individual message or URI list of messages. Again in the embodiment shown in Figure 7, the SIP REFER with INVITE request is sent directly to the network-based recycle bin bypassing the virtual user agent. Alternatively, the SIP REFER with INVITE request can be sent to the virtual user agent as described with respect to the embodiment illustrated in Figure 1.

[0038] At 740 in Figure 7, a user decides he or she wants to delete all of the messages in his or her mail storage account. At this point, the user/client device 100 sends an SIP REFER with INVITE request 750 to the recycle bin 120 (RecycleBin@mailserver.sonera.com) including the SIP URI for the user's mail storage account (in this case User@mailserver57.Sonera.com). The SIP REFER with INVITE request 750 serves to request that a SIP Session be set up between the network-based recycle bin 120 (RecycleBin@mailserver.sonera.com) and the

user's account 110 or the virtual user agent 155 if one is used. The recycle bin 120 responds by accepting the SIP REFER request from the user/client device 100 with a "202 ACCEPT" messages at 760. The recycle bin 120 also sends an INVITE request to set up a SIP session with virtual user agent 155 at 770. The virtual user agent 155 accepts this session at 780. At 790, a SIP session is officially set up with the virtual user agent 155 in the form of a message session relay protocol (MSRP), with the session description protocol (SDP) media attribute set to a=RecvOnly. The recycle bin 120 proceeds to notify the user/client 100 of the SIP session at 800, and the user/client device 100 acknowledges this notification at 810. In the SIP/MSRP sessions, all messages in the user's mail storage account (MSG1, MSG 2 and MSG3 in this case) are sent from the user's account 110 to the network-based recycle bin 120, causing all messages to disappear from the user's account 110. After successful transmission of the all of the messages from the user's mail storage account, the SIP session between the virtual user agent 155 and the recycle bin 120 is torn down. The end result, depicted at 820, is the no messages left in the user's account 110 in the user's mail storage server.

[0039] Similar to the earlier embodiments, the functions of the user's account 110 and the recycle bin 120 can also be collocated. In this situation, the sending of the INVITE requests to set up a SIP session 770, the acknowledgements of the request 780, and the setting up of the SIP session with MSRP 790 is not necessary.

[0040] Figure 8 shows a system 10 in which the present invention can be utilized, comprising multiple communication devices that can communicate through a network. The system 10 may comprise any combination of wired or wireless networks including, but not limited to, a mobile telephone network, a wireless Local Area Network (LAN), a Bluetooth personal area network, an Ethernet LAN, a token ring LAN, a wide area network, the Internet, etc. The system 10 may include both wired and wireless communication devices.

[0041] For exemplification, the system 10 shown in Figure 8 includes a mobile telephone network 11 and the Internet 28. Connectivity to the Internet 28 may include, but is not limited to, long range wireless connections, short range wireless

connections, and various wired connections including, but not limited to, telephone lines, cable lines, power lines, and the like.

[0042] The exemplary communication devices of the system 10 may include, but are not limited to, a mobile telephone 12, a combination PDA and mobile telephone 14, a PDA 16, an integrated messaging device (IMD) 18, a desktop computer 20, and a notebook computer 22. The communication devices may be stationary or mobile as when carried by an individual who is moving. The communication devices may also be located in a mode of transportation including, but not limited to, an automobile, a truck, a taxi, a bus, a boat, an airplane, a bicycle, a motorcycle, etc. Some or all of the communication devices may send and receive calls and messages and communicate with service providers through a wireless connection 25 to a base station 24. The base station 24 may be connected to a network server 26 that allows communication between the mobile telephone network 11 and the Internet 28. The system 10 may include additional communication devices and communication devices of different types.

[0043] Figures 9 and 10 show one representative mobile telephone 12 within which the present invention may be implemented. It should be understood, however, that the present invention is not intended to be limited to one particular type of mobile telephone 12 or other electronic device. Other types of electronic devices that may be used include, but are not limited to, a PDA 16, a combination PDA and mobile telephone 14, an IMD 18, a desktop computer 20, and a notebook computer 22. The communication devices may be stationary or mobile as when carried by an individual who is moving. The communication devices may also be located in a mode of transportation including, but not limited to, an automobile, a truck, a taxi, a bus, a boat, an airplane, a bicycle, a motorcycle, etc.

[0044] The mobile telephone 12 of Figures 9 and 10 includes a housing 30, a display 32 in the form of a liquid crystal display, a keypad 34, a microphone 36, an ear-piece 38, a battery 40, an infrared port 42, an antenna 44, a smart card 46 in the form of a UICC according to one embodiment of the invention, a card reader 48, radio interface circuitry 52, codec circuitry 54, a controller 56 and a memory 58. Individual circuits and elements are all of a type well known in the art, for example in the Nokia

range of mobile telephones. For exemplification, the system 10 shown in FIG. 8 includes a mobile telephone network 11 and the Internet 28. Connectivity to the Internet 28 may include, but is not limited to, long range wireless connections, short range wireless connections, and various wired connections including, but not limited to, telephone lines, cable lines, power lines, and the like.

[0045] The present invention is described in the general context of method steps, which may be implemented in one embodiment by a program product including computer-executable instructions, such as program code, executed by computers in networked environments. Generally, program modules include routines, programs, objects, components, data structures, etc. that perform particular tasks or implement particular abstract data types. Computer-executable instructions, associated data structures, and program modules represent examples of program code for executing steps of the methods disclosed herein. The particular sequence of such executable instructions or associated data structures represents examples of corresponding acts for implementing the functions described in such steps.

[0046] Software and web implementations of the present invention could be accomplished with standard programming techniques with rule based logic and other logic to accomplish the various database searching steps, correlation steps, comparison steps and decision steps. It should also be noted that the words "component" and "module," as used herein and in the claims, is intended to encompass implementations using one or more lines of software code, and/or hardware implementations, and/or equipment for receiving manual inputs.

[0047] The foregoing description of embodiments of the present invention have been presented for purposes of illustration and description. It is not intended to be exhaustive or to limit the present invention to the precise form disclosed, and modifications and variations are possible in light of the above teachings or may be acquired from practice of the present invention. The embodiments were chosen and described in order to explain the principles of the present invention and its practical application to enable one skilled in the art to utilize the present invention in various embodiments and with various modifications as are suited to the particular use contemplated.

EXHIBIT A

SIPPING Working Group
Internet-Draft
Expires: April 24, 2006

G. Camarillo
Ericsson
A. Niemi
M. Isomaki
M. Garcia-Martin
Nokia
H. Khartabil
Telio
October 21, 2005

Refering to Multiple Resources in the Session Initiation Protocol (SIP)
draft-ietf-sipping-multiple-refer-04.txt

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 April 24, 2006.

Copyright Notice

Copyright (C) The Internet Society (2005).

Abstract

This document defines extensions to the SIP REFER method so that this method can be used to refer servers to multiple resources. These extensions include the use of pointers to Uniform Resource Identifier

(URI)-lists in the Refer-To header field and the "multiple-refer" SIP option-tag.

Table of Contents

1. Introduction 3
2. Terminology 3
3. Overview of operation 3
4. The multiple-refer SIP Option-Tag 4
5. Suppressing REFER's Implicit Subscription 4
6. Behavior of SIP REFER-Issuers 5
7. Behavior of REFER-Recipients 5
8. Default URI-List Format 5
9. Example 6
10. Security Considerations 9
11. IANA Considerations 9
12. References 9
12.1. Normative References 9
12.2. Informational References 10
Authors' Addresses 11
Intellectual Property and Copyright Statements 12

1. Introduction

The SIP [3] REFER method [5] allows a user agent to request a server to send a request to a third party. Still, a number of applications need to request a server to initiate transactions towards a set of destinations. In one example, the moderator of a conference may want the conference server to send BYE requests to a group of participants. In another example, the same moderator may want the conference server to INVITE a set of new participants.

We define an extension to REFER so that REFER can be used to refer servers to multiple destinations. In addition, we use the REFER extension defined in [7] which suppresses REFER's implicit subscription.

2. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in BCP 14, RFC 2119 [1] and indicate requirement levels for compliant implementations.

We define the following three new terms:

REFER-Issuer: the user agent issuing the REFER request.
REFER-Recipient: the user agent receiving the REFER request.
REFER-Target: the intended final recipient of the request to be generated by the REFER-Recipient.

3. Overview of operation

This document defines an extension to the SIP REFER method [5] that allows a SIP User Agent Client (UAC) to include a list of REFER-Targets in a REFER request and send it to a server. The server will create a new request for each entry in the list of REFER-Target URIs.

We represent the multiple REFER-Targets of a REFER using a URI-list. A UAC (User Agent Client) that wants to refer a server to a set of destinations creates a SIP REFER request. The Refer-To header contains a pointer to a URI-list, which is included in a body part, and an option-tag in the Required header field: "multiple-refer". This option-tag indicates the requirement to support the functionality described in this specification.

When the server receives such request it creates a new request per

destination and sends them.

This document does not provide any mechanism for UACs to find out about the results of a REFER with multiple REFER-Targets. Furthermore, it does not provide support for the implicit subscription mechanism that is part of the SIP REFER method. The way UACs are kept informed about the results of a REFER is service specific. For example, a UAC sending a REFER to INVITE a set of participants to a conference may discover which participants were successfully brought into the conference by subscribing to the conference state event [9].

4. The multiple-refer SIP Option-Tag

We define a new SIP option-tag for the Require and Supported header fields: "multiple-refer".

A user agent including the "multiple-refer" option-tag in a Supported header indicates compliance with this specification.

A user agent generating a REFER with a pointer to a URI-list in its Refer-To header field MUST include the "multiple-refer" option-tag in the Require header field of the REFER.

5. Suppressing REFER's Implicit Subscription

REFER requests with a single REFER-Target establish implicitly a subscription to the refer event. The REFER-Issuer is informed about the result of the transaction towards the REFER-Target through this implicit subscription. As described in RFC 3515 [5], NOTIFY requests sent as a result of an implicit subscription created by a REFER request contain a body of type "message/sipfrag" [4] that describes the status of the transaction initiated by the REFER-Recipient.

In the case of a REFER-Issuer that generates a REFER with multiple REFER-targets, the REFER-Issuer is typically already subscribed to other event package that can provide the information about the result of the transactions towards the REFER-Targets. For example, a moderator instructing a conference server to send a BYE request to a set of participants is usually subscribed to the conference state event package for the conference. Notifications to this event package will keep the moderator and the rest of the subscribers informed of the current list of conference participants.

Most of the applications using multiple REFER do not need its implicit subscription. Consequently, a SIP REFER-Issuer generating a

REFER request with multiple REFER-Targets SHOULD include the "norefersub" option-tag in a Require header field to indicate that no notifications about the requests should be sent to the REFER-Issuer. The "norefersub" SIP option-tag is defined in [7] and suppresses the REFER's implicit subscription.

At the time of writing, there is no extension that allows to report the status of several transactions over a REFER's implicit subscription. That is the motivation for this document to recommend the usage of the "norefersub" option-tag. If in the future such an extension is defined, REFER-Issuers using it could refrain from using the "norefersub" option-tag and use the new extension instead.

6. Behavior of SIP REFER-Issuers

As indicated in Section 4 and Section 5 a SIP REFER-Issuer that creates a REFER request with multiple REFER-Targets includes a "multiple-refer" and a "norefersub" option-tags in the Require header field.

The Refer-To header field of a REFER request with multiple REFER-Targets MUST contain a pointer (i.e., a Content-ID Uniform Resource Locator (URL) [2]) that points to the body part that carries the URI-list. The REFER-Issuer SHOULD NOT include any particular URI more than once in the URI-list.

7. Behavior of REFER-Recipients

The REFER-Recipient follows the rules in Section 2.4.2 of RFC 3515 [5] to determine the status code of the response to the REFER.

If the URI-list contains a URI more than once, the REFER-Recipient MUST behave as if that URI appeared in the URI-list only once. The REFER-Recipient uses the comparison rules specific to the URI scheme of each of the URIs in the URI-list to determine if there is any URI which appears more than once.

The REFER-Recipient follows the rules in RFC 3515 [5] to generate the necessary requests towards the REFER-Targets, acting as if it had received a regular (no URI-list) REFER per each URI in the URI-list.

8. Default URI-List Format

The default format for URI-list bodies used in a multiple REFER request is the resource list document specified in [6]. User agents

able to generate or receive REFERs with multiple REFER-Targets MUST support this format as specified in [6] and MAY support other formats.

Nevertheless, the Extensible Markup Language (XML) Configuration Access Protocol (XCAP) resource list document provides features, such as hierarchical lists and the ability to include entries by reference relative to the XCAP root URI, that are not needed by the multiplet REFER service defined in this document. Therefore, when using the default resource list document, SIP REFER-Issuers generating REFERs with multiple REFER-Targets SHOULD use flat lists (i.e., no hierarchical lists) and SHOULD NOT use <entry-ref> elements.

A REFER-Recipient receiving a URI-list with more information than what has just been described MAY discard all the extra information.

Figure 1 shows an example of a flat list that follows the resource list document.

```
<?xml version="1.0" encoding="UTF-8"?>
<resource-lists xmlns="urn:ietf:params:xml:ns:resource-lists"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <list>
    <entry uri="sip:bill@example.com" />
    <entry uri="sip:joe@example.org" />
    <entry uri="sip:ted@example.net" />
  </list>
</resource-lists>
```

Figure 1: URI List

9. Example

Figure 2 shows an example flow where a REFER-Issuer sends a multiple-REFER request to the focus of a conference, which acts as the REFER-Recipient. The REFER-Recipient generates a BYE request per REFER-Target. (How to use REFER to remove participants from a conference is specified in [10].)

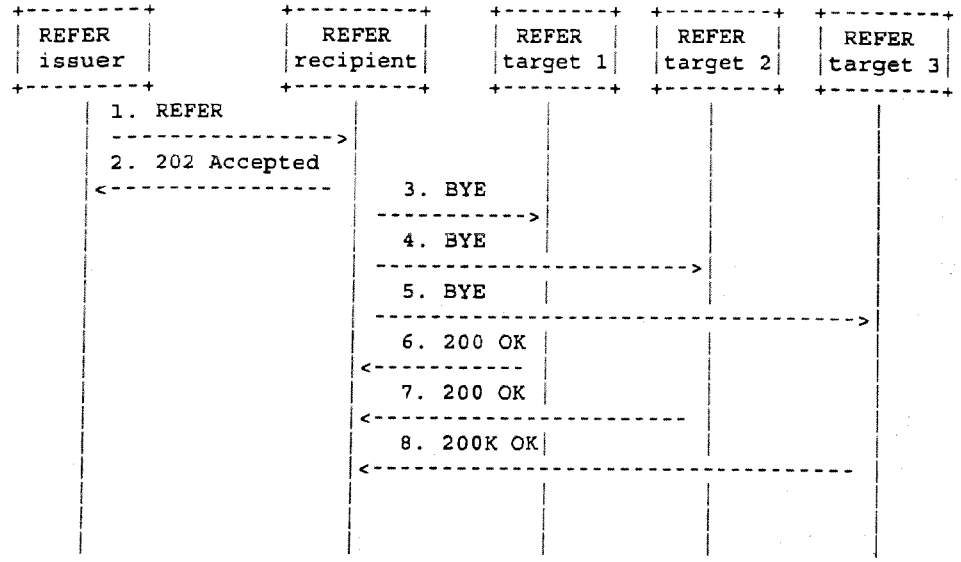


Figure 2: Example flow or a REFER request containin multiple REFER-Targets

The REFER request (1) contains a Refer-To header field that includes a pointer to the message body, which carries a list with the URIs of the REFER-Targets. The REFER's Require header field carries both the "multiple-refer" and the "norefersub" option-tags. Figure 3 shows an example of this REFER request. The resource list document contains the list of REFER-Target URIs along with the method of the SIP request that the REFER-Recipient generates.

```
REFER sip:conf-123@example.com SIP/2.0
Via: SIP/2.0/TCP client.chicago.example.com
        ;branch=z9hG4bKhjhs8ass83
Max-Forwards: 70
To: "Conference 123" <sip:conf-123@example.com>
From: Carol <sip:carol@chicago.example.com>;tag=32331
Call-ID: d432fa84b4c76e66710
CSeq: 2 REFER
Contact: <sip:carol@client.chicago.example.com>
Refer-To: <cid:cn35t8jf02@example.com>
Require: multiple-refer, norefersub
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, SUBSCRIBE, NOTIFY
Allow-Events: dialog
Accept: application/sdp, message/sipfrag
Content-Type: application/resource-lists+xml
Content-Disposition: uri-list
Content-Length: 307
Content-ID: <cn35t8jf02@example.com>

<?xml version="1.0" encoding="UTF-8"?>
<resource-lists xmlns="urn:ietf:params:xml:ns:resource-lists"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <list>
    <entry uri="sip:bill@example.com?method=BYE" />
    <entry uri="sip:joe@example.org?method=BYE" />
    <entry uri="sip:ted@example.net?method=BYE" />
  </list>
</resource-lists>
```

Figure 3: REFER request with multiple REFER-Targets

Figure 4 shows an example of the BYE request (3) that the REFER-Recipient sends to the first REFER-Target.

```
BYE sip:bill@example.com SIP/2.0
Via: SIP/2.0/TCP conference.example.com
        ;branch=z9hG4bKhjhs8assmm
Max-Forwards: 70
From: "Conference 123" <sip:conf-123@example.com>;tag=88734
To: <sip:bill@example.com>;tag=29872
Call-ID: d432fa84b4c34098s812
CSeq: 34 BYE
Content-Length: 0
```

Figure 4: BYE request

10. Security Considerations

The Framework and Security Considerations for SIP URI-List Services [8] discusses issues related to SIP URI-list services. Given that a server accepting REFERs with multiple REFER-targets acts as an URI-list service, implementations of this type of server MUST follow the security-related rules in [8]. These rules include mandatory authentication and authorization of clients, and opt-in lists.

Additionally, servers SHOULD only accept REFER requests within the context of an application the server understands (e.g., a conferencing application). This implies that servers MUST NOT accept REFERs for methods they do not understand. The idea behind these two rules is that servers are not used as dumb servers whose only function is to fan-out random messages they do not understand.

11. IANA Considerations

This document defines a new SIP option-tag: "multiple-refer". This option-tag should be registered in the SIP Parameters registry.

SIP user agents that place the "multiple-refer" option-tag in a Supported header field understand REFER requests that contain resource list document describing multiple REFER-Targets.

12. References

12.1. Normative References

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

[2] Levinson, E., "Content-ID and Message-ID Uniform Resource Locators", RFC 2392, August 1998.

[3] 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.

[4] Sparks, R., "Internet Media Type message/sipfrag", RFC 3420, November 2002.

[5] Sparks, R., "The Session Initiation Protocol (SIP) Refer Method", RFC 3515, April 2003.

[6] Rosenberg, J., "Extensible Markup Language (XML) Formats for

Representing Resource Lists",
draft-ietf-simple-xcap-list-usage-05 (work in progress),
February 2005.

[7] Levin, O., "Suppression of Session Initiation Protocol REFER Method Implicit Subscription",
draft-ietf-sip-refer-with-norefersub-03 (work in progress),
October 2005.

[8] Camarillo, G. and A. Roach, "Requirements and Framework for Session Initiation Protocol (SIP)Uniform Resource Identifier (URI)-List Services", draft-ietf-sipping-uri-services-03 (work in progress), April 2005.

12.2. Informational References

[9] Rosenberg, J., "A Session Initiation Protocol (SIP) Event Package for Conference State",
draft-ietf-sipping-conference-package-12 (work in progress),
July 2005.

[10] Johnston, A. and O. Levin, "Session Initiation Protocol Call Control - Conferencing for User Agents",
draft-ietf-sipping-cc-conferencing-07 (work in progress),
June 2005.

Authors' Addresses

Gonzalo Camarillo
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

Email: Gonzalo.Camarillo@ericsson.com

Aki Niemi
Nokia
P.O. Box 321
NOKIA GROUP, FIN 00045
Finland

Email: Aki.Niemi@nokia.com

Markus Isomaki
Nokia
Itamerenkatu 11-13
Helsinki 00180
Finland

Email: Markus.Isomaki@nokia.com

Miguel A. Garcia-Martin
Nokia
P.O.Box 407
NOKIA GROUP, FIN 00045
Finland

Email: miguel.an.garcia@nokia.com

Hisham Khartabil
Telio
P.O. Box 1203
Oslo 0110
Norway

Email: Hisham.Khartabil@telio.no

Intellectual Property Statement

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.

Disclaimer of Validity

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 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.

Copyright Statement

Copyright (C) The Internet Society (2005). 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.

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.

EXHIBIT B

SIPPING Working Group
Internet-Draft
Expires: August 27, 2006

M. Garcia-Martin
M. Isomaki
Nokia
G. Camarillo
S. Loreto
Ericsson
February 23, 2006

A Mechanism to Enable File Transfer with the Session Initiation Protocol (SIP)
draft-garcia-sipping-file-transfer-mech-00.txt

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 27, 2006.

Copyright Notice

Copyright (C) The Internet Society (2006).

Abstract

This document provides a mechanism that enables the transfer of one or more files between two User Agents (UAs). SIP and the Session Description Protocol (SDP) offer/answer model are used to signal the establishment of a session. The Message Session Relay Protocol

(MSRP) is used to actually transfer the files between the two endpoints.

Table of Contents

1. Introduction 3
2. Terminology 4
3. Definitions 4
4. Overview of Operation 5
5. Extensions to SDP 5
6. Protocol Operation 8
6.1. File selector 8
6.2. Offerer's Behavior 9
6.2.1. The Offerer is a File Sender 9
6.2.2. The Offerer is a File Receiver 9
6.2.3. SDP Offer for Several Files 10
6.3. Answerer's Behavior 10
6.3.1. The Answerer is a File Receiver 10
6.3.2. The Answerer is a File Sender 10
6.4. MSRP Usage 11
7. Examples 11
7.1. UAC sends a file to the UAS 11
7.2. UAC requests a file from the UAS 15
8. Security Considerations 17
9. IANA Considerations 17
10. Acknowledgements 17
11. References 17
11.1. Normative References 17
11.2. Informational References 18
Authors' Addresses 19
Intellectual Property and Copyright Statements 19

1. Introduction

Session Initiation Protocol (SIP) [5] provides generic functions for setting up and managing multimedia sessions between users. These sessions often contain real-time media streams such as voice and video, but are not limited to that. Basically any media component type can be supported, as long as there is a specification how to negotiate it within the Session Description Protocol (SDP) [9] offer/

answer exchange [6].

The Message Session Relay Protocol (MSRP) [10] is a protocol for transmitting instant messages (IM) in the context of a session. The protocol specification includes a description how to use it with SIP and SDP. In addition to plain text messages, MSRP is able to carry arbitrary (binary) Multipurpose Internet Mail Extensions (MIME) [2] compliant content, such as images or video clips.

There are many cases where users involved in a SIP-based multimedia session would like to exchange files within the context of that session. With MSRP it is possible to embed files as MIME objects inside the stream of instant messages. MSRP also has other features that are useful for file transfer. Message chunking enables the sharing of the same transport connection between the transfer of a large file and interactive IM exchange without blocking the IM. MSRP relays [14] provide a mechanism for Network Address Translator (NAT) traversal. Finally, Secure MIME (S/MIME) [7] can be used for ensuring the integrity and confidentiality of the transfer between the peers.

However, the baseline MSRP does not readily meet all the requirements expressed in [13] for file transfer services within SIP-based sessions. There are three main missing features:

- The recipient MUST be able to distinguish "file transfer" from "file attached to IM", allowing the recipient to treat the cases differently.
- It MUST be possible for the sender to send the request for a file transfer. It MUST be possible for the recipient to accept or decline it. The actual transfer MUST take place only after acceptance by the recipient.
- It MUST be possible for the sender to pass some meta information on the file before the actual transfer. This MUST include at least content type and size, and a short (human readable) description.

All these requirements are related to the description and negotiation of the session, not to the actual file transfer mechanism. Thus, it is natural that in order to meet them it is enough to define

attribute extensions and usage conventions to SDP, while MSRP itself needs no extensions and can be used as it is. Furthermore, all the SDP extensions can be specified in such a way that and end-point who does not support them but only implements what basic MSRP requires can still act as a Callee in a file transfer session, albeit with a somewhat reduced functionality.

This document defines the SDP attribute extensions and usage conventions needed for meeting the requirements on file transfer services within SIP sessions using MSRP as the transfer protocol within the session.

In principle it is possible to use the SDP extensions defined here and replace MSRP with any other similar protocol that can carry MIME objects. This kind of specification can be written as a separate document if the need arises.

The mechanism described in this document allows either to send or receive files to or from a remote user agent.

Section 3 defines a few terms used in this document. Section 4 provides the overview of operation. The detailed syntax and semantics of the new SDP attributes and conventions on how the existing ones are used is defined in Section 5. Section 6 describes the protocol operation involving SIP, SDP and MSRP. Some examples are given in Section 7.

2. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in BCP 14, RFC 2119 [1] and indicate requirement levels for compliant implementations.

3. Definitions

For the purpose of this document, the following definitions specified in RFC 3264 [6] apply:

- o Answerer
- o Offerer

Additionally, we define the following terms:

File sender: The endpoint that is willing to transmit a file to the file receiver.
File receiver: The endpoint that is willing to receive a file from the file sender.
File selector: The intersection of several SDP attributes that results in the selection of zero or more files. This is described in more detail in Section 6.1.

4. Overview of Operation

The file transfer service specified in this document uses the SDP offer/answer [6] to establish the MSRP-based media streams that will transfer the files. Each "m=" line describes an MSRP-based media stream used to transfer a single file. That is, the transfer of multiple files requires multiple "m=" lines.

This document defines a set of SDP attributes and some conventions that allow a user agent to describe a file to be sent to or received from a remote user agent. This way, user agents can decide whether or not to accept a given file transfer based on the file's name, size, description, hash, icon (e.g., if the file is a picture), etc.

Effectively, the aim of this mechanism is similar to the aim of the content indirection mechanism in SIP [15]. Both mechanisms allow a user agent to decide whether or not to download a file based on information about the file. However, while the content indirection mechanism could be used in SIP MESSAGE [12] requests to transfer files (the actual file transfer mechanism would be the Hypertext Transfer Protocol (HTTP) [11]), it cannot be used within a session established by an offer/answer exchange where the file transfer protocol is MSRP [10].

5. Extensions to SDP

We define a number of attributes in SDP [9] that provide the required information to describe the transfer of a file with MSRP. The following is the formal ABNF syntax [8] of these new attributes. It is built above the SDP [9] grammar, RFC 2045 [2], and RFC 2392 [3].


```
attribute          = filename-attr / filetype-attr /
                     disposition-attr / filesize-attr /
                     icon-attr / hash-attr
                     ;attribute is defined in sdp-new

filename-attr      = "filename:" filename-string
filename-string    = byte-string    ;byte-string defined in sdp-new

filetype-attr      = "filetype:" type "/" subtype *(";" parameter)
                          ; parameter defined in RFC 2045
type               = token
subtype            = token

disposition-attr   = "disposition:" disposition-value
disposition-value  = token

filesize-attr      = "filesize:" filesize-value
filesize-value     = integer        ;integer defined in sdp-new

icon-attr          = "icon:" icon-value
```

```
icon-value          = cid-url          ;cid-url defined in RFC 2392

hash-attr           = "hash:" hash-algorithm WSP hash-value
hash-algorithm      = token            ;see IANA Hash Algorithm
                                       ;section in the IPSEC
                                       ;registry
hash-value          = byte-string      ;byte-string defined in sdp-new
```

Figure 1: Syntax of the SDP extension

The 'filename' attribute contains the filename of the content, and its value is a byte string (specified in SDP [9]).

The 'filetype' attribute contains the MIME media type of the content. In general, anything that can be expressed in a Content-Type header field (see RFC 2045 [2]) can also be expressed with the 'filetype' attribute. Possible MIME Media Type values are the ones listed in the IANA registry for MIME Media Types. Zero or more parameters can follow. The syntax of 'parameter' is specified in RFC 2045 [2].

The 'disposition' attribute provides a suggestion to the peer of the intended disposition of the file. Possible values are the one listed in the IANA registry for Mail Content Disposition Values, although most likely only the "inline" and "attachment" values are significant for file transfer applications.

The 'filesize' attribute indicates the size of the file in octets.

The 'icon' attribute can be useful with certain file types such as images. It allows the sender to include a pointer to a body that includes an icon representing the contents of the file to be transferred. This allows the sender to include the icon as another body accompanying the SDP, and to the recipient to get the icon of the file that can potentially be transferred. It is recommended to keep icons restricted to the minimum number of bytes that provide significance. The 'icon' attribute contains a Content-ID URL, which is specified in RFC 2392 [3].

The 'hash' attribute provides a hash of the file to be transferred. The purpose is two-fold: On one side, it allows the file receiver to identify a file by its hash rather than by its file name, providing that the file receiver has learn the hash of the file by some out-of-band mechanism. On the other side, it allows the file sender to provide the hash of the file to be transmitted, which can be used by the file receiver for verification of its contents or to avoid the unnecessary transmission of a file that already exists. The 'hash' attribute includes the type of hash and its value. Possible types of hash are the ones defined in the Hash Algorithm Section of the IANA registry of the IPSec registry. Implementations according to this specification MUST implement the US Secure Hash Algorithm 1 (SHA1) [4] and MAY implement other hashing algorithms. The creator of the SDP MAY also add more than one 'hash' attribute (presumably with

different types of hash) to the same file. The value is the byte string resulting of applying the hash algorithm to the content of the file.

The following is an example of an SDP body that contains the extensions defined in this memo:

```
v=0
o=alice 2890844526 2890844526 IN IP4 host.atlanta.example.com
s=
c=IN IP4 host.atlanta.example.com
t=0 0
m=message 7654 TCP/MSRP *
i=This is my latest picture
a=sendonly
a=accept-types:*
a=path:msrp://atlanta.example.com:7654/jshA7we;tcp
a=filename:My cool picture.jpg
a=filetype:image/jpeg
a=disposition:inline
a=filesize:32349
a=icon:cid:id2@alicepc.example.com
a=hash:SHA 72245fe8653ddaf371362f86d471913ee4a2ce2e
```


Figure 2: Example of SDP describing a file transfer

6. Protocol Operation

This Section discusses how to use the parameters defined in Section 5 in the context of an offer/answer [6] exchange. Additionally, this section also discusses the behavior of the endpoints using MSRP.

Usually the file transfer session is initiated when the offerer sends an SDP offer to the answerer. The answerer either accepts or rejects the file transfer session and sends an SDP answer to the offerer.

We can differentiate two use cases, depending on whether the offerer is the file sender or file receiver:

1. The offerer is the file sender, i.e., the offerer wants to transmit a file to the answerer. The answerer is the file receiver. In this case the SDP offer contains a 'sendonly' attribute, and accordingly the SDP answer contains a 'recvonly' attribute.
2. The offerer is the file receiver, i.e., the offerer wants to fetch a file from the answerer. The answerer is the file sender. In this case the SDP offer contains a 'recvonly' attribute, and accordingly the SDP answer contains a 'sendonly' attribute.

6.1. File selector

The protocol specified in this document requires a mechanism to identify files in a remote host. We introduce the concept of a file selector, which is the intersection of the 'hash', 'filename', 'filesize', and 'filetype' attributes.

A file selector can point to zero, one, or more files, depending on the presence of the mentioned attributes in the SDP and depending on the available files in a host. The file transfer mechanism that we specify in this document requires that a file selector results in a single file selection. Typically, if the 'hash' attribute is known, the 'hash' attribute is enough to produce a file selector that points to zero or one file. However, not always the file selector is known or available. Sometimes only the 'filename', 'filesize', or 'filetype' attributes are known, so the file selector may result in more than one file, an undesired case. The opposite is also true, if the file selector contains a 'hash' and a 'filename' attributes, but the user at the remote host has renamed the file, although there is a file with the indicated hash, the file name does not match, thus, the file selector will result in the selection of zero files.

6.2. Offerer's Behavior

An offerer that wishes to send or receive one or more files to or from an answerer MUST build an SDP [9] description of a session containing one or more "m=" lines, each one describing an MSRP session (and thus, one file transfer operation), according to the MSRP [10] procedures. All the media line attributes specified and required by MSRP [10] (e.g., "a=path", "a=accept-types", etc.) MUST be included as well. For each file to be transferred there MUST be a separate "m=" line.

6.2.1. The Offerer is a File Sender

If the offerer is a file sender, it MUST add a session or media 'sendonly' attribute to the SDP offer. Additionally, the offerer SHOULD also add a 'filetype', 'filesize', and 'hash' attributes indicating the type, size, and hash of the file, respectively.

> These attributes might not be known when the offerer creates the SDP offer, for example, because the host is still processing the file.
>
> The 'hash' attribute contains valuable information to the file receiver to identify whether the file is already available and need not be transmitted.

The offerer MAY also add a 'filename', 'icon', and 'disposition' attributes further describing the file to be transferred. The 'disposition' attribute provides a presentation suggestion, (for example: the file sender would like the file receiver to render file "inline", or save it as an "attachment"). Additionally, the offerer MAY provide a human readable description of the file using an "i="

media line.

6.2.2. The Offerer is a File Receiver

If the offerer is a file receiver, it MUST create an SDP offer that contains a session or media 'recvonly' attribute. Then the offerer SHOULD add at least one of the attributes that constitute the file selector ('hash', 'filename', 'filesize', or 'filetype'). In many cases, if the hash of the file is known, that is enough to identify the file, therefore, the offerer can include only a 'hash' attribute. However, specially in cases where the hash of the file is unknown, the file name, size, and type can provide a description of the file to be fetched.

6.2.3. SDP Offer for Several Files

An offerer that wishes to send or receive more than one file generates an "m=" line per file. This way, the answerer can reject individual files by setting the port number of their associated "m=" lines to zero, as per regular SDP [9] procedures.

Using an "m=" line per file implies that different files are transferred using different MSRP sessions. However, all those MSRP sessions can be set up to run over a single TCP connection, as described in Section 8.1 of [10].

6.3. Answerer's Behavior

If the answerer wishes to reject a file offered by the offerer, it sets the port number of the "m=" line associated with the file to zero, as per regular SDP [9] procedures. If the answerer decides to accept the file, it proceeds as per regular MSRP [10] and SDP [9] procedures.

6.3.1. The Answerer is a File Receiver

If the answerer is a file receiver and decides to accept the file transfer it MUST create an SDP answer (per RFC 3264 [6]) containing a 'recvonly' attribute. If the offer contains 'filetype', 'filesize', or 'filename' attributes, the answerer MUST copy them into the answer. This informs the offerer that the answerer supports this specification. If the answerer is a file receiver, it MUST NOT include 'icon', 'hash', or 'disposition' attributes in the SDP answer.

If the received offer contains a 'hash' attribute, the answerer can use it to find out if a local file with the same hash is already available, in which case, this could imply the reception of a duplicated file. It is up to the answerer to determine whether the file transfer is accepted or not in case of a duplicated file.

6.3.2. The Answerer is a File Sender

If the answerer is a file sender, it MUST first inspect the received SDP offer and compute the file selector. The file selector is the result of a intersection of the 'filetype', 'filesize', 'filename', and 'hash' attributes (if they are present) that modify the same "m=" line in the SDP offer (i.e., the four mentioned attributes are located under the same "m=" line in SDP). The file selector identifies zero or more files to be sent. If the file selector is unable to identify any file, then the answerer MUST reject the MSRP stream for file transfer by setting the port number to zero (and if

it is the only stream in the SDP offer, then if SHOULD reject the SDP as per procedures in RFC 3264 [6]).

If the file selector points to a single file and the answerer decides to accept the file transfer, the answerer MUST create an SDP answer (per RFC 3264 [6]) that contains a 'sendonly' attribute. The answerer SHOULD add a 'hash' attribute containing the hash of the file to be sent and MAY include 'filename', 'filetype', 'filesize', 'icon', or 'disposition' attributes to further describe the file.

Last, if the file selector points to multiple files, the answerer SHOULD reject the MSRP media stream for file transfer (by setting the port number to zero).

If the need arises, future specifications can provide a suitable mechanism that allows to either select multiple files or, e.g., resolve ambiguities by returning a list of files that match the file selector.

6.4. MSRP Usage

The file transfer service specified in this document uses "m=" lines to describe the unidirectional transfer of a file. Consequently, each MSRP session established following the procedures in Section 6.2 and Section 6.3 is only used to transfer a single file. So, senders MUST only use a given MSRP session to send the file described in the SDP offer or answer. That is, senders MUST NOT send additional files over the same MSRP session.

Once the file transfer is completed, the file sender SHOULD close the MSRP session, and MUST behave according to the MSRP [10] procedures with respect closing MSRP sessions.

7. Examples

7.1. UAC sends a file to the UAS

This section shows an example flow for a file transfer scenario. The example assumes that SIP is used to transport the SDP exchange,

although the SIP details are briefly shown in the sake of brevity.

Alice, the SDP offerer, wishes to send an image file to Bob (the answerer). Alice's UAC creates a unidirectional SDP offer that contains the description of the file that she wants to send to Bob. The description also includes an icon representing the contents of the file to be transferred. The sequence flow is shown in Figure 3.

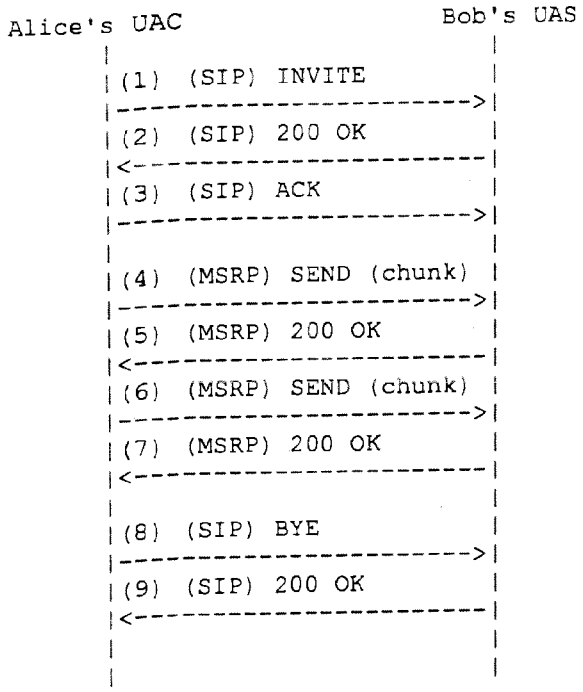


Figure 3: Flow diagram of a UAC sending a file to a UAS

F1: Alice constructs an SDP description of the file to be sent and attaches it to a SIP INVITE request addressed to Bob.

```
INVITE sip:bob@example.com SIP/2.0
To: Bob <sip:bob@example.com>
From: Alice <sip:alice@example.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 1 INVITE
Max-Forwards: 70
Date: Fri, 17 Feb 2006 13:02:03 GMT
Contact: <sip:alice@alicepc.example.com>
Content-Type: multipart/mixed; boundary="boundary71"
Content-Length: [length]

--boundary71
Content-Type: application/sdp
Content-Length: [length of SDP]

v=0
o=alice 2890844526 2890844526 IN IP4 alicepc.example.com
s=
c=IN IP4 alicepc.example.com
t=0 0
m=message 7654 TCP/MSRP *
i=This is my latest picture
a=sendonly
a=accept-types:*
a=path:msrp://alicepc.example.com:7654/jshA7we;tcp
a=filename:My cool picture.jpg
a=filetype:image/jpeg
a=disposition:inline
a=filesize:4096
a=icon:cid:id2@alicepc.example.com
a=hash:SHA 72245fe8653ddaf371362f86d471913ee4a2ce2e

--boundary71
Content-Type: image/jpeg
Content-Transfer-Encoding: binary
Content-ID: <id2@alicepc.example.com>
Content-Length: [length of image]

...binary JPEG image...
--boundary71
```

From now on we omit the SIP details for the sake of brevity.

F2: Bob receives the INVITE request, inspects the SDP offer and extracts the icon body, checks the file size and decides to accept the file transfer. So Bob creates the following SDP answer:

```
v=0
o=bob 2890844656 2890844656 IN IP4 bobpc.example.com
s=
c=IN IP4 bobpc.example.com
t=0 0
m=message 8888 TCP/MSRP *
a=recvonly
a=accept-types:*
a=path:msrp://bobpc.example.com:8888/9di4ea;tcp
a=filename:My cool picture.jpg
a=filetype:image/jpeg
a=filesize:4096
```

F4: Alice opens a TCP connection to Bob and creates an MSRP SEND request. This SEND request contains the first chunk of the file.

```
MSRP d93kswow SEND
To-Path: msrp://bobpc.example.com:8888/9di4ea;tcp
From-Path: msrp://alicepc.example.com:7777/iau39;tcp
Message-ID: 12339sdqwer
Byte-Range: 1-2048/4096
Content-Type: image/jpeg

... first chunk of the JPEG image ...
-------d93kswow+
```

F5: Bob acknowledges the reception of the first chunk.

```
MSRP d93kswow 200 OK
To-Path: msrp://alicepc.example.com:7777/iau39;tcp
From-Path: msrp://bobpc.example.com:8888/9di4ea;tcp
Byte-Range: 1-2048/4096
-------d93kswow$
```

F6: Alice sends the second and last chunk.

```
MSRP op2nc9a SEND
To-Path: msrp://bobpc.example.com:8888/9di4ea;tcp
From-Path: msrp://alicepc.example.com:7777/iau39;tcp
Message-ID: 12339sdqwer
Byte-Range: 2049-4096/4096
Content-Type: image/jpeg

... second (and last) chunk of the JPEG image ...
-------op2nc9a$
```

F5: Bob acknowledges the reception of the second chunk.

```
MSRP op2nc9a 200 OK
To-Path: msrp://alicepc.example.com:7777/iau39;tcp
From-Path: msrp://bobpc.example.com:8888/9di4ea;tcp
Byte-Range: 2049-4096/4096
-------op2nc9a$
```

F8: Alice terminates the SIP session by sending a SIP BYE request.

F9: Bob acknowledges the reception of the BYE request and sends a 200 (OK) response.

7.2. UAC requests a file from the UAS

In this example Alice, the SDP offerer, wishes to fetch a file from Bob, the SDP answerer. Alice knows that Bob has a specific file she wants to download. She has learned the hash of the file by some out-of-band mechanism. The hash attribute is enough to produce a file selector that points to the specific file. So, Alice creates an SDP offer that contains the file descriptor. Bob accepts the transmission and sends the file to Alice. Figure 10 shows the sequence flow.

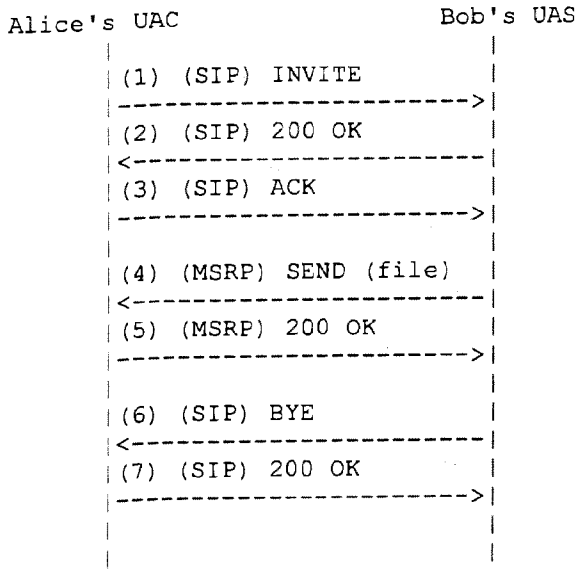


Figure 10: Flow diagram of a UAC requesting a file from a UAS

F1: Alice constructs n SDP description of the file she wants to receive and attaches the SDP offer to a SIP INVITE request addressed to Bob.

```
INVITE sip:bob@example.com SIP/2.0
To: Bob <sip:bob@example.com>
From: Alice <sip:alice@example.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 1 INVITE
Max-Forwards: 70
Date: Fri, 17 Feb 2006 13:02:03 GMT
Contact: <sip:alice@alicepc.example.com>
Content-Type: application/sdp
Content-Length: [length of SDP]

v=0
o=alice 2890844526 2890844526 IN IP4 alicepc.example.com
s=
c=IN IP4 alicepc.example.com
t=0 0
m=message 7654 TCP/MSRP *
a=recvonly
a=accept-types:image/jpeg
a=path:msrp://alicepc.example.com:7654/jshA7we;tcp
a=hash:SHA 72245fe8653ddaf371362f86d471913ee4a2ce2e
```

From now on we omit the SIP details for the sake of brevity.

F2: Bob receives the INVITE request, inspect the SDP offer, computes the file descriptor and finds a local file whose hash equals the one indicated in the SDP. Bob accepts the file transmission and creates an SDP answer as follows:

```
v=0
o=bob 2890844656 2890844656 IN IP4 bobpc.example.com
s=
c=IN IP4 bobpc.example.com
t=0 0
m=message 8888 TCP/MSRP *
a=sendonly
a=accept-types:*
a=path:msrp://bobpc.example.com:8888/9di4ea;tcp
a=filename:My cool photo.jpg
a=filetype:image/jpeg
a=filesize:2027
```

F4: Alice opens a TCP connection to Bob. Bob then creates an MSRP SEND request that contains the file.


```
MSRP d93kswow SEND
To-Path: msrp://alicepc.example.com:7777/iau39;tcp
From-Path: msrp://bobpc.example.com:8888/9di4ea;tcp
Message-ID: 12339sdqwer
```

```
Byte-Range: 1-2027/2027
Content-Type: image/jpeg

...binary JPEG image...
-------d93kswow$
```

F6: Alice acknowledges the reception of the SEND request.

```
MSRP d93kswow 200 OK
To-Path: msrp://bobpc.example.com:8888/9di4ea;tcp
From-Path: msrp://alicepc.example.com:7777/iau39;tcp
Byte-Range: 1-2027/2027
-------d93kswow$
```

F6: Then Bob terminates the SIP session by sending a SIP BYE request.

F7: Alice acknowledges the reception of the BYE request and sends a 200 (OK) response.

8. Security Considerations

TBD

9. IANA Considerations

TBD

10. Acknowledgements

The authors would like to thank Mats Stille, Nancy Greene, Adamu Haruna, and Arto Leppisaari for discussing initial concepts described in this memo. Thanks to Pekka Kuure for reviewing initial versions this document and providing helpful comments.

11. References

11.1. Normative References

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

[2] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.

[3] Levinson, E., "Content-ID and Message-ID Uniform Resource Locators", RFC 2392, August 1998.

[4] Eastlake, D. and P. Jones, "US Secure Hash Algorithm 1 (SHA1)",

RFC 3174, September 2001.

[5] 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.

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

[7] Ramsdell, B., "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification", RFC 3851, July 2004.

[8] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 4234, October 2005.

[9] Handley, M., "SDP: Session Description Protocol", draft-ietf-mmusic-sdp-new-25 (work in progress), July 2005.

[10] Campbell, B., "The Message Session Relay Protocol", draft-ietf-simple-message-sessions-13 (work in progress), December 2005.

11.2. Informational References

[11] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[12] Campbell, B., Rosenberg, J., Schulzrinne, H., Huitema, C., and D. Gurle, "Session Initiation Protocol (SIP) Extension for Instant Messaging", RFC 3428, December 2002.

[13] Isomaki, M., "Requirements and Possible Mechanisms for File Transfer Services Within the Context of SIP Based Communication", draft-isomaki-sipping-file-transfer-00 (work in progress), October 2005.

[14] Jennings, C., "Relay Extensions for the Message Sessions Relay Protocol (MSRP)", draft-ietf-simple-msrp-relays-06 (work in

progress), December 2005.

[15] Burger, E., "A Mechanism for Content Indirection in Session Initiation Protocol (SIP) Messages", draft-ietf-sip-content-indirect-mech-05 (work in progress), October 2004.

Authors' Addresses

Miguel A. Garcia-Martin
Nokia

P.O.Box 407
NOKIA GROUP, FIN 00045
Finland

Email: miguel.an.garcia@nokia.com

Markus Isomaki
Nokia
Keilalahdentie 2-4
Espoo 02150
Finland

Email: markus.isomaki@nokia.com

Gonzalo Camarillo
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

Email: Gonzalo.Camarillo@ericsson.com

Salvatore Loreto
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

Email: Salvatore.Loreto@ericsson.com

Full Copyright Statement

Copyright (C) The Internet Society (2006).

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 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 currently provided by the Internet Society.

WHAT IS CLAIMED IS:

1. A method of deleting an item from a user account in a SIP multimedia environment, comprising:

receiving a request from a user device to delete the item from the user account;

establishing a SIP session between a network-based deleted items location and the user account;

after establishing the SIP session, transferring the item from the user account to the network-based deleted items location; and

deleting the item from the user account.

2. The method of claim 1, wherein the request comprises an SIP REFER request.

3. The method of claim 2, wherein the SIP REFER request includes an address for the network-based deleted items location in its Refer-to header.

4. The method of claim 1, wherein the request comprises a SIP Multiple-REFER request.

5. The method of claim 1, further comprising, in response to receiving the request, transmitting an acknowledgment of the request to the user device.

6. The method of claim 1, wherein the SIP session is established with a SDP directional attribute set to a=SendOnly.

7. The method of claim 1, wherein the SIP session is established with a SDP directional attribute set to a=RecvOnly

8. The method of claim 1, wherein the item includes a unique message identifier, and wherein the unique message identifier is included in the request from the user device.

9. The method of claim 1, wherein the request is to delete multiple items from the user account and wherein a URI-list of the multiple items is included in the request from the user device.

10. The method of claim 1, wherein the request is to delete all items from the user account and wherein a SIP URI for the user account is included in the request from the user account.

11. The method of claim 1, wherein the item is stored as a file having SDP descriptions and wherein the file SDP descriptions are included in the request from the user device.

12. The method of claim 11 wherein the request is to delete multiple items from the user account and wherein file SDP descriptions for each of the multiple items is included in the request from the user device the file SDP descriptions for each file being included in a separate media line in the request.

13. The method of claim 1, wherein the user device, the user's account and the network-based deleted items location each possess a unique uniform resource identifier.

14. The method of claim 1, wherein MSRP protocol is used to transfer the item from the user account to the network-based deleted items location.

15. A computer program product, embodied in a computer-readable medium, for deleting an item from a user account in a SIP multimedia environment, comprising:

computer code for receiving a request from a user device to delete the item from the user account;

computer code for establishing a SIP session between a network-based deleted items location and the user account;

computer code for after establishing the SIP session, transferring the item from the user account to the network-based deleted items location; and

computer code for deleting the item from the user account.

16. The computer program product of claim 15, wherein the request comprises an SIP REFER request.

17. The computer program product of claim 16, wherein the SIP REFER request includes an address for the network-based deleted items location in its Refer-to header.

18. The computer code product of claim 15, wherein the request comprises a SIP Multiple-REFER request.

19. The computer program product of claim 15, wherein the SIP session is established with a SDP directional attribute set to a=SendOnly.

20. The computer program product of claim 15, wherein the SIP session is established with a SDP directional attribute set to a=RecvOnly.

21. The computer program product of claim 15, wherein the item includes a unique message identifier, and wherein the unique message identifier is included in the request from the user device.

22. The computer program product of claim 15, wherein the request is to delete multiple items from the user account and wherein a URI-list of the multiple items is included in the request from the user device.

23. The computer program product of claim 15, wherein the request is to delete all items from the user account and wherein a SIP URI for the user account is included in the request from the user account.

24. The computer program product of claim 15, wherein the item is stored as a file having SDP descriptions and wherein the SDP descriptions are included in the request from the user device.

25. The computer program product of claim 23, wherein the request is to delete multiple items from the user account and wherein file SDP descriptions for

each of the multiple items is included in the request from the user device the file SDP descriptions for each file being included in a separate media line in the request.

26. The computer program product of claim 15, wherein the user device, the user's account and the network-based deleted items location each possess a unique uniform resource identifier.

27. The computer program product of claim 15, wherein MSRP protocol is used to transfer the item from the user account to the network-based deleted items location.

28. An electronic device, comprising:

a processor; and

a memory unit communicatively connected to the processor and including:

computer code for receiving an SIP request from a user device to delete an item from a user account;

computer code for establishing a SIP session between a network-based deleted items location and the user account;

computer code for, after establishing the SIP session, transferring the item from the user account to the network-based deleted items location; and

computer code for deleting the item from the user account.

29. The electronic device of claim 28, wherein the request comprises an SIP REFER request.

30. The electronic device of claim 29, wherein the SIP REFER request includes an address for the network-based deleted items location in its Refer-to header.

31. The electronics device of claim 28 wherein the request comprises a SIP Multiple-REFER request.

32. The electronic device of claim 28, wherein the SIP session is established with a SDP directional attribute set to a=SendOnly.

33. The electronic device of claim 28, wherein the SIP session is established with a SDP directional attribute set to a=RecvOnly.

34. The electronic device of claim 28, wherein the item includes a unique message identifier, and wherein the unique message identifier is included in the request from the user device.

35. The electronic device of claim 28, wherein the request is to delete multiple items from the user account and wherein a URI-list of the multiple items is included in the request from the user device.

36. The electronic device of claim 28, wherein the request is to delete all items form the user account and wherein a SIP URI for the user account is included in the request from the user device.

37. The electronic device of claim 28, wherein the item is stored as a file having SDP descriptions and wherein the file SDP descriptions are included in the request from the user device.

38. The electronic device of claim 37, wherein the request is to delete multiple items from the user account and wherein file SDP descriptions for each of the multiple items is included in the request from the user device, the file SDP descriptions for each file being included in a separate media line in the request.

39. The electronic device of claim 28, wherein MSRP protocol is used to transfer the item from the user account to the network-based deleted items location.

40. A method of deleting an item from a user account in a SIP multimedia environment, comprising:

transmitting a request from a user device to delete the item from the user account;

in response to the transmitted request establishing a SIP session between a virtual user agent and a network-based deleted items location;

after establishing the SIP session, transferring the item from the user account to the network-based deleted items location; and

deleting the item from the user account.

41. The method of claim 40, wherein the SIP session is established with a SDP directional attribute set to a=SendOnly.

42. The method of claim 40, wherein the SIP session is established with a SDP directional attribute set to a=RecvOnly.

43. The method of claim 40, wherein the request comprises a SIP REFER request including a unique identifier for the item to be deleted.

44. The method of claim 40, wherein the request is to delete multiple items, the request comprising a SIP Multiple-REFER request including a URI list of the items to be deleted.

45. The method of claim 40, wherein the request is to delete all items, the request comprising a SIP REFER request including a SIP URI for the user account.

46. The method of claim 40, wherein the item is stored as a file having SDP descriptions and wherein the request comprises a REFER request including the SDP descriptions.

47. The method of claim 40, wherein items are stored as files having SDP descriptions, wherein the request is to delete multiple items and wherein the file SDP descriptions of the items to be deleted are included in the request from the user device, the file SDP descriptions for each file being included in a separate media line in the request.

48. The method of claim 40, wherein the SIP REFER request includes an address for the network-based deleted items location in its Refer-to header.

49. The method of claim 40, wherein MSRP protocol is used to transfer the item from the user account to the network-based deleted items location.

50. A method of deleting an item from a user account in a SIP multimedia environment, comprising:

transmitting a SIP REFER request from a user device to delete the item from the user account, the SIP REFER request including an address for a network-based deleted items location in its Refer-to header;

in response to receiving the SIP REFER request, transmitting an acknowledgment of the request to the user device; and

deleting the item from the user account.