A Transaction Event Package for the Session Initiation Protocol (SIP)
draft-camarillo-sipping-transac-package-00.txt

Statue of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026.

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 6, 2004.

Copyright Notice

Copyright (C) The Internet Society (2004). All Rights Reserved.

Abstract

SIP provides a SIP Events notification framework that is extensible through the addition of event packages. This document defines a transaction event package for the SIP Events notification, along with a data format used in notifications for this package. The transaction package allows users to subscribe to a resource in an application server and receive notifications about the changes in state of transactions the application server initiates as part of a service. Additionally, we define a new SIP Associated-Transactions-State header field that allows a server to return a subscribable URI that provides transactions notification information.
Table of Contents

1. Introduction ................................................. 3
2. Terminology .................................................. 3
3. Definitions ................................................... 3
4. The Transaction Event Package ................................. 4
4.1 Event Package Name ........................................ 4
4.2 Event Package Parameters ................................... 4
4.3 SUBSCRIBE Bodies ........................................... 4
4.4 Subscription Duration ....................................... 4
4.5 NOTIFY Bodies ............................................... 4
4.6 Notifier Processing of SUBSCRIBE Requests .................. 4
4.7 Notifier Generation of NOTIFY Requests ..................... 5
4.8 Subscriber Processing of NOTIFY Requests ................... 5
4.9 Handling of Forked Requests ................................ 5
4.10 Rate of Notifications ...................................... 5
4.11 State Agents ............................................... 5
5. Transaction Information Format ................................ 5
5.1 Structure of the Transaction Information .................... 6
5.1.1 Transaction Element ..................................... 6
5.2 Constructing Coherent State ................................ 7
5.3 Schema ...................................................... 8
5.4 Example ..................................................... 9
6. The Associated-Transactions-State Header Field ............... 10
6.1 Syntax ...................................................... 10
7. Security Considerations ..................................... 11
8. IANA Considerations ......................................... 11
8.1 MIME Registration for application/transaction-info+xml .... 11
8.2 URN Sub-Namespace Registration for urn:ietf:params:xml:ns:transaction-info ..................... 12
8.3 Schema Registration ....................................... 12
8.4 Associated-Transactions-State Header Field Registration .. 13
9. Acknowledgements ............................................. 13
10. Normative References ..................................... 13
11. Informational References .................................. 13
12. Authors’ Addresses ......................................... 14
13. Intellectual Property and Copyright Statements ............ 15
1. Introduction

The SIP [5] Events framework [6] defines general mechanisms for subscription to, and notification of, events within SIP networks. It introduces the notion of a package, which is a specific "instantiation" of the events mechanism for a well-defined set of events. Here, we define an event package for transactions.

An example of an application using this package is a MESSAGE exploder (SIP exploders are described in [8]) that sends MESSAGE requests to a set of destinations on behalf of a user. The user subscribes to the transaction state of the transactions generated by the application server as part of the service. The subscriber receives one or more notifications containing the status of those transactions. This way, the user is informed of which MESSAGE requests were delivered to their destination and which ones failed.

The transaction state of the transactions generated by the application server as part of the service is identified by a URI. The user agent uses this URI to subscribe to this state. The user agent may use different mechanisms to obtain such a URI. Section 6 defines one of such mechanisms: the Associated-Transactions-State SIP header field.

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

We define the following terms:

Triggering transaction: A SIP transaction that triggers a set of actions in an application server. These actions usually include the generation of one or more SIP associated transactions by the application server.

Triggering request: The SIP request that is part of a triggering transaction.

Associated transaction: A SIP transaction that is generated by an application server on reception of a triggering request.
4. The Transaction Event Package

This section provides the details for defining a SIP Events package, as specified by RFC 3265 [6].

4.1 Event Package Name

The name of this event package is "transaction". This package name is carried in the Event and Allow-Events header fields, as defined in [6].

4.2 Event Package Parameters

This package does not define any event package parameters.

4.3 SUBSCRIBE Bodies

This package does not define any SUBSCRIBE bodies.

4.4 Subscription Duration

The default subscription duration for this event package is 60 seconds.

4.5 NOTIFY Bodies

In this event package, the body of the notification contains a transaction information document. This document describes the state of one or more transactions associated with the subscribed resource. All subscribers and notifiers MUST support the "application/transaction-info+xml" data format described in Section 5. The subscribe request MAY contain an Accept header field. If no such header field is present, it has a default value of "application/transaction-info+xml". If the header field is present, it MUST include "application/transaction-info+xml", and MAY include any other types capable of representing transaction state.

The notifications generated by the server MUST be in one of the formats specified in the Accept header field in the SUBSCRIBE request.

4.6 Notifier Processing of SUBSCRIBE Requests

The transaction information for a resource contains sensitive information. So, all subscriptions SHOULD be authenticated. Authorization policy is at the discretion of the administrator of the notifier.
4.7 Notifier Generation of NOTIFY Requests

The notifier MUST generate a notification containing the state of all the transactions associated with the subscribed resource as soon as any of the following actions take place: a) all the transactions complete; b) the subscription timer expires.

This behaviour guarantees that the subscriber gets at least one notification as soon as the transactions are complete or, within the subscription timer.

The notifier MAY send notifications more often (e.g., once every time the state of a transaction changes) if there are filters applied to the subscription.

4.8 Subscriber Processing of NOTIFY Requests

On reception of a valid NOTIFY request, the subscriber SHOULD immediately render the transaction status to the end-user in an implementation specific way.

4.9 Handling of Forked Requests

By their nature, the resources supported by this package are centralized. So, SUBSCRIBE requests should not generally fork. Users of this package MUST NOT install more than a single subscription as a result of a single SUBSCRIBE request.

4.10 Rate of Notifications

For reasons of congestion control, it is important that the rate of notifications not become excessive. As a result, it is RECOMMENDED that the server not generate notifications for a single subscriber at a rate faster than once every five seconds.

4.11 State Agents

Transaction state is ideally maintained in the element which generates the transactions. Consequently, the elements that generate the transactions are the ones best suited to handle subscriptions to it. The usage of state agents is NOT RECOMMENDED for this package.

5. Transaction Information Format

Transaction information is an XML document that MUST be well-formed and SHOULD be valid. Transaction information documents MUST be based on XML 1.0 and MUST be encoded using UTF-8. This specification makes use of XML namespaces for identifying Transaction information.
documents. The namespace URI for elements defined by this specification is a URN [2], using the namespace identifier ‘ietf’ defined by RFC 2648 [3] and extended by [7]. This URN is:

`urn:ietf:params:xml:ns:transaction-info`

A Transaction information document begins with the root element tag "transaction-info".

5.1 Structure of the Transaction Information

A transaction information document starts with a "transaction-info" element. This element has three mandatory attributes:

- **version**: allows the recipient of transaction information documents to properly order them. Versions start at 0, and increment by one for each new document sent to a subscriber. Versions are scoped within a subscription. Versions MUST be representable using a 32 bit integer.

- **state**: indicates whether the document contains the "full" transaction information, or whether it contains only information on those transactions which have changed since the previous document ("partial").

- **entity**: contains a URI that identifies the resource whose transaction information is reported in the remainder of the document.

The "transaction-info" element has a series of zero or more "transaction" sub-elements.

5.1.1 Transaction Element

The "transaction" element contains information on a particular associated transaction. It has two mandatory attributes: "id" and "r-uri".

- **id**: provides a single string that can be used as an identifier for this transaction. The "id" is created when the request that initiates the transaction is sent and it MUST be unique amongst all the transactions at the subscribed resource.

- **r-uri**: provides the Request-URI of the request that initiated the associated transaction.

The transaction element has a mandatory sub-element: the "state" sub-element.
5.1.1.1 State Element

The "state" element indicates the state of the transaction. Its value is an enumerated type describing one of the following two states: "pending" or "complete". It has an optional "code" attribute that contains a provisional response status code (in the pending state) or a final response code (in the complete state).

The following is an example of a state element:

    <state code="404">complete</state>

5.2 Constructing Coherent State

The subscriber to the transaction information maintains a table for the list of transactions. The table contains a row for each transaction. Each row is indexed by an ID, present in the "id" attribute of the "transaction" element. The contents of each row contain the state of that transaction as conveyed in the document. The table is also associated with a version number. The version number MUST be initialized with the value of the "version" attribute from the "transaction-info" element in the first document received. Each time a new document is received, the value of the local version number, and the "version" attribute in the new document, are compared. If the value in the new document is one higher than the local version number, the local version number is increased by one, and the document is processed. If the value in the document is more than one higher than the local version number, the local version number is set to the value in the new document, and the document is processed. If the document did not contain full state, the subscriber SHOULD generate a refresh request to trigger a full state notification. If the value in the document is less than the local version, the document is discarded without processing. The "transaction-info" element contains an "entity" attribute that indicate the URI of the subscribed resource.

The processing of the transaction information document depends on whether it contains full or partial state. If it contains full state, indicated by the value of the "state" attribute in the "transaction-info" element, the contents of the table are flushed. They are repopulated from the document. A new row in the table is created for each "transaction" element. If the document contains partial state, as indicated by the value of the "state" attribute in the "transaction-info" element, the document is used to update the table. For each "transaction" element in the document, the subscriber checks to see whether a row exists for that transaction. This check
is done by comparing the ID in the "id" attribute of the "transaction" element with the ID associated with the row. If the transaction does not exist in the table, a row is added, and its state is set to the information from that "transaction" element. If the transaction does exist, its state is updated to be the information from that "transaction" element. If a row is updated or created, such that its state is now terminated, that entry MAY be removed from the table at any time.

5.3 Schema

The following is the schema for the application/transaction-info+xml type:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="urn:ietf:params:xml:ns:transaction-info"
    xmlns:tns="urn:ietf:params:xml:ns:transaction-info"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified">
    <!-- This import brings in the XML language attribute xml:lang--> 
        schemaLocation="http://www.w3.org/2001/03/xml.xsd"/>
    <xs:element name="transaction-info">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="tns:transaction"
                    maxOccurs="unbounded"/>
                <xs:any namespace="##other" processContents="lax"
                    minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attribute name="version" type="xs:nonNegativeInteger"
                use="required"/>
            <xs:attribute name="state" use="required">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                        <xs:enumeration value="full"/>
                        <xs:enumeration value="partial"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="entity" type="xs:anyURI"
                use="required"/>
        </xs:complexType>
    </xs:element>
</xs:schema>
```
5.4 Example

The following is an example of a application/transaction-info+xml document:

```xml
<?xml version="1.0"?>
<transaction-info xmlns="urn:ietf:params:xml:ns:transaction-info"
    version="0"
    state="full"
    entity="sip:explosion44@exploder12.example.com">
    <transaction id="frgd870th87" r-uri="sip:bob@example.com">
        <state code="200">complete</state>
    </transaction>
    <transaction id="234f12345" r-uri="sip:alice@example.com">
        <state code="404">complete</state>
    </transaction>
    <transaction id="fghd2345" r-uri="sip:mary@example.com">
        <state>pending</state>
    </transaction>
</transaction-info>
```
6. The Associated-Transactions-State Header Field

User agents may need to obtain a URI that identifies a set of transactions at the application server in order to subscribe to the state of those transactions. A user agent can use different means to obtain such a URI. One of them consists of using the Associated-Transactions-State SIP header field, which we define here.

The Associated-Transactions-State header field can be used when an application server receives a SIP request that causes the application server to initiate a set of transactions. We refer to this SIP request as the triggering request and the set of transactions as the associated transactions.

For example, an application server provides conferencing services. When this application server receives an INVITE from a user who is the first joining a particular conference, the application server sends a MESSAGE to the rest of the participants to inform them that there is already a user in the conference. In this case, the triggering request is the INVITE, and the associated transactions are the MESSAGE transactions.

Application servers MAY include a Associated-Transactions-State header field in the responses to a triggering request. Clients can use the URI in this header field to subscribe to the state of the associated transactions.

In our example, the response to the INVITE request carries a Associated-Transactions-State header field. The client subscribes to the URI received in this header field to monitor the state of the MESSAGE transactions. This way, the user knows who of the rest of the participants receives the MESSAGE.

6.1 Syntax

The ABNF of the Associated-Transactions-State header field is:

```
Associated-Transactions-State = HCOLON transactions-state-uri
transactions-state-uri = SIP-URI / SIPS-URI
```
OPEN ISSUE: do we want to have transactions-state-uri = addr-spec, which includes SIP-URI / SIPS-URI / absoluteURI? Do we need non-SIP URIs?

7. Security Considerations

TBD.

8. IANA Considerations

This document registers a new MIME type, application/transaction-info+xml, new XML namespace and a new SIP header field.

8.1 MIME Registration for application/transaction-info+xml

MIME media type name: application

MIME subtype name: transaction-info+xml

Mandatory parameters: none

Optional parameters: Same as charset parameter application/xml as specified in RFC 3023 [4].

Encoding considerations: Same as encoding considerations of application/xml as specified in RFC 3023 [4].

Security considerations: See Section 10 of RFC 3023 [4] and Section 7 of this specification.

Interoperability considerations: none.

Published specification: This document.

Applications which use this media type: This document type has been used to support SIP applications such as MESSAGE exploders.

Additional Information:

Magic Number: None

File Extension: .tin or .xml

Macintosh file type code: "TEXT"

Personal and email address for further information: Gonzalo Camarillo, <Gonzalo.Camarillo@ericsson.com>
8.2 URN Sub-Namespace Registration for
urn:ietf:params:xml:ns:transaction-info

This section registers a new XML namespace, as per the guidelines in [7].

URI: The URI for this namespace is

Registrant Contact: IETF, SIPPING working group,<sipping@ietf.org>,
Gonzalo Camarillo, <Gonzalo.Camarillo@ericsson.com>

XML:

BEGIN
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML Basic 1.0//EN"
"http://www.w3.org/TR/xhtml-basic/xhtml-basic10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="content-type"
content="text/html; charset=iso-8859-1"/>
<title>Transaction Information Namespace</title>
</head>
<body>
<h1>Namespace for Transaction Information</h1>
<h2>application/transaction-info+xml</h2>
<p>See <a href="[[URL of published RFC]]">RFCXXX</a>.</p>
</body>
</html>
END

8.3 Schema Registration

This specification registers a schema, as per the guidelines in [7].

URI: please assign.

Registrant Contact: IETF, SIPPING working group,<sipping@ietf.org>,
Gonzalo Camarillo, <Gonzalo.Camarillo@ericsson.com>

XML: The XML can be found in Section 5.3.
8.4 Associated-Transactions-State Header Field Registration

9. Acknowledgements

This document is partially based on the event package for INVITE initiated dialogs [9].

Normative References


Informational References


Authors’ Addresses

Gonzalo Camarillo
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

EMail: Gonzalo.Camarillo@ericsson.com

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

EMail: miguel.an.garcia@nokia.com
Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any intellectual property 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; neither does it represent that it has made any effort to identify any such rights. Information on the IETF’s procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. Copies of claims of rights made available for publication 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 implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

Full Copyright Statement

Copyright (C) The Internet Society (2004). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assignees.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION
Acknowledgment

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