ALTO Incremental Updates Using Server-Sent Events (SSE)

draft-ietf-alto-incr-update-sse-17

W. Roome
Y. Richard Yang

IETF 104
March 26, 2019
Prague

Updates Overview (v13-v17)

- https://www.ietf.org/rfcdiff?url1=draft-ietf-alto-incr-update-sse-13&url2=draft-ietf-alto-incr-update-sse-17
- Two major types of changes
  - Fix issues identified in WGLC reviews – most are organizing and clarification issues
  - Fix issues motivated by handling PV – a systematic approach to handling multipart/related
- Huge group efforts by many, in several weekly meetings

Update: Updated Draft Structure

Update: Highlight of Major Changes

```
2.  Major Changes Since Version -01

   To RFC editor: This will be removed in the final version.  We keep
   this section to make clear major changes in the technical content.

   o  Incremental encoding: Added JSON patch as an alternative
      incremental delta encoding.

   o  Update concurrent requests of the same resource: The client now
      assigns a unique client-id to each resource in an update stream.
      The server puts the client-id in each update event for that

      resource (before, the server used the server's resource-id).  This
      allows a client to use one update stream to get updates to two
      different requests with the same server resource-id; before, that
      required two separate update streams.

   o  Control: Defined a new "stream control" resource (Section 8) to
      allow clients to add or remove resources from a previously created
      update stream.  The ALTO server creates a new stream control
      resource for each update stream instance, assigns a unique URI to
      it, and sends the URI to the client as the first event in the
      stream.
```

```
2.  Major Changes Since Version -01

   To RFC editor: This will be removed in the final version.  We keep
   this section to make clear major changes in the technical content.

   o  Incremental encoding using JSON patch: Added JSON patch as an
      alternative incremental delta encoding than Merge patch.

   o  Substream-id to allow concurrent updates of the same server
      resource: This design allows an ALTO client to assign a unique
      substream-id when requesting a resource in an update stream.  The
      server puts the substream-id in each update event for that
      resource (before, the server used the server's resource-id).  This
      allows a client to use one update stream to receive updates to
      multiple requests for the same server resource, for example, for a
      POST-mode resource with different input parameters; before, that
      required separate update streams.

   o  Multipart resources: Use generic `data-id` subfield of the `event`
      field to identify the data to be updated.  For all major existing
      data, data-id is the substream-id, but it allows support of
      multipart as well, by adding content-id.

   o  Flexible control: Defined a new "stream control" resource
      (Section 8) to allow a client to add or remove resources from a
      previously created update stream.  The ALTO server creates a new
      stream control resource for each update stream instance, assigns a
      unique URI to it, and sends the URI to the client as the first
      event in the stream.
```

Update: Restructure the Order of Terms

3. Terms

This document uses the following terms: Update Stream, Update Message, Data Update Message, Full Replacement, Incremental Change, Update Stream Server, Update Stream Control Service, Update Stream Control, Control Update Message, Stream Control Service.

Update Stream: An update stream is an HTTP connection between an ALTO client and an ALTO server so that the server can push a sequence of update messages using SSE to the client.

Update Message: An update message is either a data update message or a control update message.

Data Update Message: A data update message is for a single ALTO information resource and sent from the update stream server to the ALTO client when the resource changes. A data update message can be either a full-replacement message or an incremental-change message. Full replacement is a shorthand for a full-replacement message, and incremental change is a shorthand for an incremental-change message.

Full Replacement: A full replacement for a resource encodes the content of the resource in its original ALTO encoding.

Incremental Change: An incremental change specifies only the difference between the new content and the previous version. An incremental change can be encoded using either JSON merge patch or JSON patch in this document.

Update Stream Server: An update stream server is an ALTO server that provides update stream service.

Stream Control Service: An stream control service provides an HTTP URI so that the ALTO client of an update stream can use it to request the addition or removal of resources receiving update messages.

Stream Control: A shorthand for stream control service.

Control Update Message: A control update message of an update stream is for the update stream server to notify the ALTO client of related control information of the update stream. The first control update message provides the URI using which the ALTO client can send stream control requests to the stream control server. Additional control update messages allow the update stream server to notify the ALTO client of status changes (e.g., the server will no longer send updates for an information resource).

3. Terms

This document uses the following terms: Update Stream, Update Stream Server, Update Message, Data Update Message, Full Replacement, Incremental Change, Control Update Message, Stream Control Service, Stream Control.

Update Stream: An update stream is an HTTP connection between an ALTO client and an ALTO server so that the server can push a sequence of update messages using SSE to the client.

Update Stream Server: We refer to an ALTO server providing an update stream as an ALTO update stream server, or update stream server for short. Note that the ALTO server mentioned in this document refers to a general server that provides various kinds of services; it can be an update stream server or stream control server (see below); it can also be a server providing ALTO IRD information.

Update Message: An update message is either a data update message or a control update message.

Data Update Message: A data update message is for a single ALTO information resource and sent from the update stream server to the ALTO client when the resource changes. A data update message can be either a full-replacement message or an incremental-change message. Full replacement is a shorthand for a full-replacement message, and incremental change is a shorthand for an incremental-change message.

Full Replacement: A full replacement for a resource encodes the content of the resource in its original ALTO encoding.

Incremental Change: An incremental change specifies only the difference between the new content and the previous version. An incremental change can be encoded using either JSON merge patch or JSON patch in this document.

Stream Control Service: An stream control service provides an HTTP URI so that the ALTO client of an update stream can use it to send stream control requests on the addition or removal of resources receiving update messages from the update stream.

Stream Control: A shorthand for stream control service.

Stream Control Server: An stream control server providing the stream control service.

Control Update Message: A control update message is a message in an update stream for the update stream server to notify the ALTO client of related control information of the update stream. The first message of an update stream is a control update message and provides the URI using which the ALTO client can send stream control requests to the stream control server. Additional control update messages in an update stream allow the update stream server to notify the ALTO client of status changes (e.g., the server will no longer send updates for an information resource).

Update: Clarify Interaction

stream server to send server state updates to the ALTO client as well, showing as control update messages from the update stream server to the ALTO client.

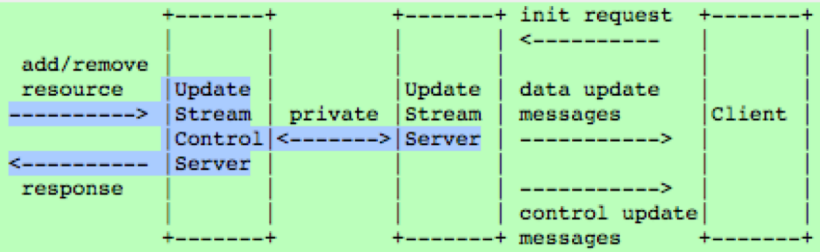


Figure 2: ALTO SSE Overview

In addition to state changes triggered from the update stream server side, in a flexible design, an ALTO client may initiate changes as well, in particular, by adding or removing ALTO resources receiving updates. An ALTO client initiates such changes using the stream control service. For an update stream service supporting update stream control, the update stream server responds by sending an event (a control update message) with the URI of the stream control service. The ALTO client can then use the URI to ask the update stream server to (1) send data update messages for additional resources, (2) stop sending data update messages for previously requested resources, or (3) gracefully stop and close the update stream altogether.

stream server to send server control updates (vs data updates) to the ALTO client as well, showing as control update messages from the update stream server to the ALTO client.

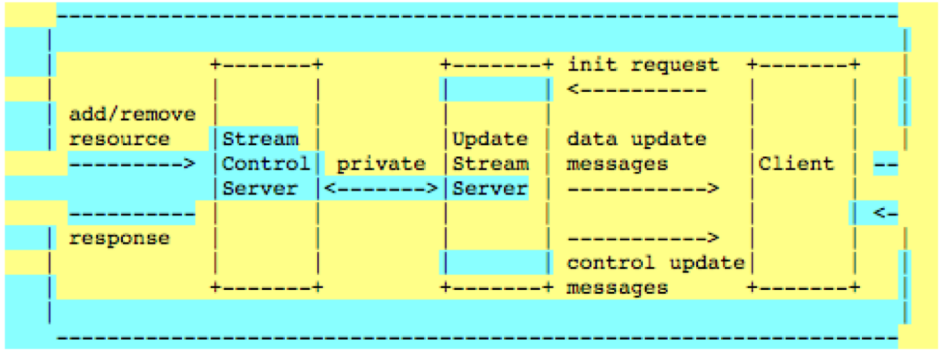
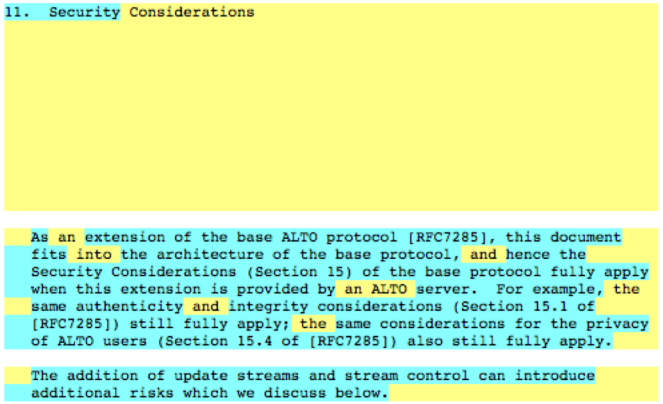


Figure 2: ALTO SSE Architecture.

In addition to control changes triggered from the update stream server side, in a flexible design, an ALTO client may initiate control changes as well, in particular, by adding or removing ALTO resources receiving updates. An ALTO client initiates such changes using the stream control service. Although one may use a design that the client uses the same HTTP connection to send the control requests, it requires stronger server support such as HTTP pipeline. For more flexibility, this document introduces stream control service. In particular, the update stream server of an update stream uses the first message to provide the URI of the stream control service. The ALTO client can then use the URI to ask the update stream server to (1) send data update messages for additional resources, (2) stop sending data update messages for previously requested resources, or (3) gracefully stop and close the update stream altogether. Figure 2 shows the complete ALTO SSE architecture.

Update: Clarify Security Considerations

- Clarify original security issues still apply



11. Security Considerations

As an extension of the base ALTO protocol [RFC7285], this document fits into the architecture of the base protocol, and hence the Security Considerations (Section 15) of the base protocol fully apply when this extension is provided by an ALTO server. For example, the same authenticity and integrity considerations (Section 15.1 of [RFC7285]) still fully apply; the same considerations for the privacy of ALTO users (Section 15.4 of [RFC7285]) also still fully apply.

The addition of update streams and stream control can introduce additional risks which we discuss below.

- Categorize security issues into three aspects: stream server, ALTO client and stream control server

Update: Provide Better Wording on Alternative Design

```
   object-map {
      ClientId -> AddUpdateReq;
   } AddUpdatesReq;

   object {
      String       resource-id;
      [String      tag;]
      [Boolean     incremental-changes;]
      [Object      input;]
   } AddUpdateReq;
```

add: specifies the resources for which the ALTO client wants updates, and has one entry for each resource. An ALTO client creates a unique client-id (Section 6.1) for each such resource, and uses those client-ids as the keys in the "add" field.

resource-id: the resource-id of an ALTO resource, and MUST be in the update stream's "uses" list (Section 8.5.2 of Section 7.5). If the resource-id is a GET-mode resource with a version tag (or "vtag"), as defined in Section 6.3 and Section 10.3 of [RFC7285], and the ALTO client has previously retrieved a version of that resource from the update stream server, the ALTO client MAY set the "tag" field to the tag part of the client's version of that resource. If that version is not current, the update stream server MUST send a full replacement before sending any incremental changes, as described in Section 7.7.1. If that version is still current, the update stream server MAY omit the initial full replacement.

incremental-changes: the ALTO client specify whether it is willing to receive incremental changes from the update stream server for a specific resource-id. If the "incremental-changes" field for a resource-id is "true", the update stream server MAY send incremental changes for that resource (assuming the update stream server supports incremental changes for) that resource; see Section Section 7.4). If the "incremental-changes" field is "false", the update stream server MUST NOT send incremental changes for that resource. The default value for "incremental-changes" is "true", so to suppress incremental changes, the ALTO client MUST explicitly set "incremental-changes" to "false".

Note that the ALTO client cannot suppress full replacement.

```
   object-map {
      SubstreamId -> AddUpdateReq;
   } AddUpdatesReq;

   object {
      String       resource-id;
      [String      tag;]
      [Boolean     incremental-changes;]
      [Object      input;]
   } AddUpdateReq;
```

add: specifies the resources (and the parameters for the resources) for which the ALTO client wants updates. We say that the add-request creates a substream. The ALTO client MUST assign a unique substream-id (Section 6.1) for each entry, and uses those substream-ids as the keys in the "add" field.

resource-id: the resource-id of an ALTO resource, and MUST be in the update stream's "uses" list (Section 8.5.2 of Section 7.5). If the resource-id is a GET-mode resource with a version tag (or "vtag"), as defined in Section 6.3 and Section 10.3 of [RFC7285], and the ALTO client has previously retrieved a version of that resource from the update stream server, the ALTO client MAY set the "tag" field to the tag part of the client's version of that resource. If that version is not current, the update stream server MUST send a full replacement before sending any incremental changes, as described in Section 7.7.1. If that version is still current, the update stream server MAY omit the initial full replacement.

incremental-changes: the ALTO client specifies whether it is willing to receive incremental changes from the update stream server for this substream. If the "incremental-changes" field is "true", the update stream server MAY send incremental changes for this substream (assuming the update stream server supports incremental changes for) that resource; see Section Section 7.4). If the "incremental-changes" field is "false", the update stream server MUST NOT send incremental changes for that substream. The default value for "incremental-changes" is "true", so to suppress incremental changes, the ALTO client MUST explicitly set "incremental-changes" to "false". An alternative design of incremental-changes control is a more fine-grained control, by allowing a client to select the subset of incremental methods from the set announced in the server's capabilities (see Section Section 7.4). But this adds complexity to server, which is more likely to be the bottleneck.

Note that the ALTO client cannot suppress full replacement.

Integrated Handling of Multipart

6. Update Messages: Data Update and Control Update Messages

We now define the details of ALTO incremental update. Specifically, an update stream consists of a stream of data update messages (Section 6.2) and control update messages (Section 6.3).

6.1. ALTO Update Message Format

Data update and control update messages have the same basic structure. The data field is a JSON object, and the event field contains the media type of the data field and an optional client-id. Data update messages use the client-id to identify the ALTO resource to which the update message applies. Client-ids MUST follow the rules for ALTO ResourceIds (Section 10.2 of [RFC7285]). Client-ids MUST be unique within an update stream, but need not be globally unique. For example, if an ALTO client requests updates for both a cost map and its dependent network map, the ALTO client might assign client-id "1" to the network map and client-id "2" to the cost map. Alternatively, the ALTO client could use the client-ids for those two maps.

JSON specifications use the type ClientId for a client-id, and the type ClientId conforms to the specification of ResourceId as defined in Section 10.2 of [RFC7285].

The two sub-fields (media-type and client-id) of the event field are encoded as comma-separated strings:

media-type [',' client-id]

Note that media-type names may not contain a comma (character code 0x2c). [Dawn: may not or MAY NOT]

6. Update Messages: Data Update and Control Update Messages

We now define the details of ALTO SSE. Specifically, an update stream consists of a stream of data update messages (Section 6.2) and control update messages (Section 6.3).

6.1. ALTO Update Message Format

Data update and control update messages have the same basic structure: each message includes a data field to provide data information, which is typically a JSON object; and an event field preceding the data field, to specify the media type indicating the encoding of the data field.

A data update message needs additional information to identify the ALTO data to which the update message applies. For example, an ALTO client can request updates for both a cost map and its dependent network map in the same update stream. The ALTO client assigns substream-id "1" in its request to receive updates to the network map; and substream-id "2" to the cost map. For this example, the substream-id defines the data to be updated and need to be indicated in a data update message.

Hence, the event field of ALTO update message can include two sub-fields (media-type and data-id), where the two sub-fields are separated by a comma:

media-type [',' data-id]

To allow non-ambiguous decoding of the two sub-fields, the media-type name used by ALTO SSE MUST NOT contain a comma (character code 0x2c), and the string before the comma is the media-type name. [To RFC editor: please check this conforms to Section 4.2 of [RFC6838] and confirms to IANA.]

Integrated Handling of Multipart

6.2. ALTO Data Update Message

A data update message is sent when a monitored resource changes. The data is either a complete specification of the resource, or else a patch (either JSON merge patch or JSON patch) describing the changes from the last version. We will refer to these as full replacement and incremental change, respectively. The encoding of full replacement is defined by [RFC7285]; examples are network and cost map messages. They have the media types defined in that document. The encoding of JSON merge patch is defined by [RFC7396], with media type "application/merge-patch+json"; the encoding of JSON patch is defined by [RFC6902], with media type "application/json-patch+json".

6.2. ALTO Data Update Message

A data update message is sent when a monitored resource changes. In [RFC7285], each resource is encoded as a single JSON object. In the general case, a resource may include multiple JSON objects. This document considers the case that a resource may contain multiple components (parts) and they are encoded using multipart/related [RFC2387]. Each component requiring the service of this document MUST be identified by a unique Content-ID to be defined in its defining document.

The `data-id` sub-field identifies the ALTO data to which a data update message applies. For a resource containing only a single JSON object, the substream-id assigned by the client when requesting the SSE service is enough to identify the data. In this document, substream-ids MUST follow the rules for ALTO ResourceIds (Section 10.2 of [RFC7285]). Substream-ids MUST be unique within an update stream, but need not be globally unique. For a resource using multipart/related, the `data-id` sub-field must include the substream-id, `.` and the unique Content-ID.

A data update is either a complete specification of the identified data, or else an incremental patch (e.g., a JSON merge patch or JSON patch), if possible, describing the changes from the last version of the data. This document refers to these as full replacement and incremental change, respectively. The encoding of a full replacement is defined its defining document (e.g., network and cost map messages by [RFC7285], and uses media type defined in that document. The encoding of JSON merge patch is defined by [RFC7396], with media type "application/merge-patch+json"; the encoding of JSON patch is defined by [RFC6902], with media type "application/json-patch+json".

Summary: Status

- The revision has addressed all issues identified so far
- Plan:
  - Take a pass to address any WG comments, if raised: in 1 week after IETF
  - Work through the rest of the process