      Calendaring Extensions to WebDAV (CalDAV): Managed Attachments

Abstract

   This specification adds an extension to the Calendaring Extensions to
   WebDAV (CalDAV) to allow attachments associated with iCalendar data
   to be stored and managed on the server.

   This specification documents existing code deployed by multiple
   vendors. It is published as an Informational specification rather
   than Standards Track due to its noncompliance with multiple best
   current practices of HTTP.

Status of This Memo

   This document is not an Internet Standards Track specification; it is
   published for informational purposes.

   This document is a product of the Internet Engineering Task Force
   (IETF). It represents the consensus of the IETF community. It has
   received public review and has been approved for publication by the
   Internet Engineering Steering Group (IESG). Not all documents
   approved by the IESG are candidates for any level of Internet
   Standard; see Section 2 of RFC 7841.

   Information about the current status of this document, any errata,
   and how to provide feedback on it may be obtained at
   https://www.rfc-editor.org/info/rfc8607.

Copyright Notice

Table of Contents

1. Introduction

   The iCalendar [RFC5545] data format is used to represent calendar
   data and is used with iCalendar Transport-independent
   Interoperability Protocol (iTIP) [RFC5546] to handle scheduling
   operations between calendar users.

   [RFC4791] defines the Calendaring Extensions to WebDAV (CalDAV),
   based on HTTP [RFC7230], for accessing calendar data stored on a
   server.

   Calendar users often want to include attachments with their calendar
   data events or tasks (for example, a copy of a presentation or the
   meeting agenda). iCalendar provides an "ATTACH" property whose value
   is either the inline Base64 encoded attachment data or a URL
   specifying the location of the attachment data.

   Use of inline attachment data is not ideal with CalDAV because the
   data would need to be uploaded to the server each time a change to
   the calendar data is made, even minor changes such as a change to the
   summary. Whilst a client could choose to use a URL value instead,
   the problem then becomes where and how the client discovers an
   appropriate URL to use and how it ensures that only those attendees
   listed in the event or task are able to access it.

   This specification solves this problem by having the client send the
   attachment to the server, separately from the iCalendar data, and
   having the server add appropriate "ATTACH" properties in the
   iCalendar data as well as manage access privileges. The server can
   also provide additional information to the client about each
   attachment in the iCalendar data, such as the size and an identifier.

1.1. Rationale for Informational Status

   Although this extension to CalDAV has wide deployment, its design
   does not comply with some of the best current practices of HTTP,
   namely:

   o All operations on attachments are modeled as HTTP POST operations,
      where the actual type of operation is specified using a query
      parameter instead of using separate HTTP POST, PUT, and DELETE
      methods where appropriate.

   o Specific query strings are hardwired into the protocol in
      violation of Section 2.4 of [RFC7320].

   Additionally, this extension misuses the Content-Disposition header
   field [RFC6266] as a request header field to convey a filename for an
   attachment rather than using an existing request header field
   suitable for that purpose, such as "Slug" (see Section 9.7 of
   [RFC5023]).

   Rather than creating interoperability problems with deployed code by
   updating the design of this extension to be compliant with best
   current practices and to allow this specification to be placed on the
   Standards Track, it was decided to simply document how existing
   implementations interoperate and to publish the document as
   Informational.

2. Conventions Used in This Document

   The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
   "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and
   "OPTIONAL" in this document are to be interpreted as described in
   BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all
   capitals, as shown here.

   The notation used in this memo is the ABNF notation of [RFC5234] as
   used by iCalendar [RFC5545]. Any syntax elements shown below that
   are not explicitly defined in this specification come from iCalendar
   [RFC5545].

3. Overview

   There are four main operations a client needs to perform with
   attachments for calendar data: add, update, remove, and retrieve.
   The first three operations are carried out by the client issuing an
   HTTP POST request on the calendar object resource to which the
   attachment is associated and specifying the appropriate "action"
   query parameter (see Section 3.3). In the case of the remove

operation, the client can alternatively directly update the calendar
object resource and remove the relevant "ATTACH" properties (see
Section 3.9). The retrieve operation is accomplished by simply
issuing an HTTP GET request targeting the attachment URI specified by
the calendar resource's "ATTACH" property (see Section 3.10).

iCalendar data stored in a CalDAV calendar object resource can
contain multiple components when recurrences are involved. In such a
situation, the client needs to be able to target a specific
recurrence instance or multiple instances when adding or deleting
attachments. As a result, the POST request needs to provide a way
for the client to specify which recurrence instances should be
targeted for the attachment operation. This is accomplished through
use of additional query parameters on the POST Request-URI.

3.1. Requirements

   A server that supports the features described in this specification
   is REQUIRED to support the CalDAV "calendar-access" [RFC4791]
   features.

   In addition, such a server SHOULD support the "return=representation"
   Prefer header field [RFC7240] preference on successful HTTP PUT and
   POST requests targeting existing calendar object resources by
   returning the new representation of that calendar resource (including
   its new ETag header field value) in the response.

3.2. Discovering Support for Managed Attachments

   A server supporting the features described in this specification MUST
   include "calendar-managed-attachments" as a token in the DAV response
   header field (as defined in Section 10.1 of [RFC4918]) from an
   OPTIONS request on a calendar home collection.

   A server might choose not to support the storing of managed
   attachments on a per-recurrence-instance basis (i.e., they can only
   be added to all instances as a whole). If that is the case, the
   server MUST also include "calendar-managed-attachments-no-recurrence"
   as a token in the DAV response header field from an OPTIONS request
   on a calendar home collection. When that field is present, clients
   MUST NOT attempt any managed attachment operations that target
   specific recurrence instances.

3.3. POST Request for Managing Attachments

   An HTTP POST request is used to add, update, or remove attachments.
   These requests are subject to the preconditions listed in
   Section 3.11. The Request-URI will contain various query parameters
   to specify the behavior.

3.3.1. action Query Parameter

   The "action" query parameter is used to identify which attachment
   operation the client is requesting. This parameter MUST be present
   once on each POST request used to manage attachments. One of these
   three values MUST be used:

   attachment-add: Indicates an operation that is adding an attachment
      to a calendar object resource. See Section 3.4 for more details.

   attachment-update: Indicates an operation that is updating an
      existing attachment on a calendar object resource. See
      Section 3.5 for more details.

   attachment-remove: Indicates an operation that is removing an
      attachment from a calendar object resource. See Section 3.6 for
      more details.

      Example:

      https://calendar.example.com/events/1.ics?action=attachment-add

3.3.2. rid Query Parameter

   The "rid" query parameter is used to identify which recurrence
   instances are being targeted by the client for the attachment
   operation. This query parameter MUST contain one or more items,
   separated by commas (denoted in ASCII as "0x2C"). The item values
   can be in one of two forms:

   Master instance: The value "M" (case insensitive) refers to the
      "master" recurrence instance, i.e., the component that does not
      include a "RECURRENCE-ID" property. This item MUST be present
      only once.

   Specific instance: A specific iCalendar instance is targeted by
      using its "RECURRENCE-ID" value as the item value. That value
      MUST correspond to the "RECURRENCE-ID" value as stored in the
      calendar object resource (i.e., without any conversion to UTC).
      If multiple items of this form are used, they MUST be unique
      values. For example, to target a recurrence defined by property

      RECURRENCE-ID;TZID=America/Montreal:20111022T160000, the query
      parameter rid=20111022T160000 would be used.

   If the "rid" query parameter is not present, all recurrence instances
   in the calendar object resource are targeted.

   The "rid" query parameter MUST NOT be present in the case of an
   update operation, or if the server chooses not to support per-
   recurrence instance managed attachments (see Section 3.2).

   Example (targeting the master instance and a specific overridden
   instance):

   https://calendar.example.com/events/1.ics?
     action=attachment-add&rid=M,20111022T160000

3.3.3. managed-id Query Parameter

   The "managed-id" query parameter is used to identify which "ATTACH"
   property is being updated or removed. The value of this query
   parameter MUST match the "MANAGED-ID" (Section 4.3) property
   parameter value on the "ATTACH" property in the calendar object
   resource instance(s) targeted by the request.

   The "managed-id" query parameter MUST NOT be present in the case of
   an add operation.

   Example:

   https://calendar.example.com/events/1.ics?
     action=attachment-update&managed-id=aUNhbGVuZGFy

3.4. Adding Attachments

   To add an attachment to an existing calendar object resource, the
   following needs to occur:

   1. The client issues a POST request targeted at the calendar object
       resource structured as follows:

       A. The Request-URI will include an "action" query parameter with
           the value "attachment-add" (see Section 3.3.1).

       B. If all recurrence instances are having an attachment added,
           the "rid" query parameter is not present in the Request-URI.
           If one or more specific recurrence instances are targeted,
           then the Request-URI will include a "rid" query parameter
           containing the list of instances (see Section 3.3.2).

   C. The body of the request contains the data for the attachment.

   D. The client MUST include a valid Content-Type header field
       describing the media type of the attachment (as required by
       HTTP).

   E. The client SHOULD include a Content-Disposition header field
       [RFC6266] with a "type" parameter set to "attachment", and a
       "filename" parameter that indicates the name of the
       attachment. Note that the use of Content-Disposition as a
       request header field is nonstandard and specific to this
       protocol.

   F. The client MAY include a Prefer header field [RFC7240] with
       the "return=representation" preference to request that the
       modified calendar object resource be returned as the body of
       a successful response to the POST request.

2. When the server receives the POST request, it does the following:

   A. Validates that any recurrence instances referred to via the
       "rid" query parameter are valid for the calendar object
       resource being targeted.

   B. Stores the supplied attachment data into a resource and
       generates an appropriate URI for clients to access the
       resource.

   C. For each affected recurrence instance in the calendar object
       resource targeted by the request, adds an "ATTACH" property
       whose value is the URI of the stored attachment. The
       "ATTACH" property MUST contain a "MANAGED-ID" property
       parameter whose value is a unique identifier (within the
       context of the server as a whole). The "ATTACH" property
       SHOULD contain an "FMTTYPE" property parameter whose value
       matches the Content-Type header field value from the request.
       The "ATTACH" property SHOULD contain a "FILENAME" property
       parameter whose value matches the value of the Content-
       Disposition header field "filename" parameter value from the
       request, taking into account the restrictions expressed in
       Section 4.2. The "ATTACH" property SHOULD include a "SIZE"
       property parameter whose value represents the size in octets
       of the attachment. If a specified recurrence instance does
       not have a matching component in the calendar object
       resource, then the server MUST modify the calendar object
       resource to include an overridden component with the
       appropriate "RECURRENCE-ID" property.

   D. Upon successful creation of the attachment resource, and
       modification of the targeted calendar object resource, it
       MUST return an appropriate HTTP success status response and
       include a "Cal-Managed-ID" header field containing the
       "MANAGED-ID" property parameter value of the newly created
       "ATTACH" property. The client can use the "Cal-Managed-ID"
       header field value to correlate the attachment with "ATTACH"
       properties added to the calendar object resource. If the
       client included a Prefer header field with the
       "return=representation" preference in the request, the server
       SHOULD return the modified calendar object resource as the
       body of the response. Otherwise, the server can expect that
       the client will reload the calendar object resource with a
       subsequent GET request to refresh any local cache.

   In the following example, the client adds a new attachment to a
   nonrecurring event and asks the server (via the Prefer header field
   [RFC7240]) to return the modified version of that event in the
   response.

   >> Request <<

   POST /events/64.ics?action=attachment-add HTTP/1.1
   Host: cal.example.com
   Content-Type: text/html; charset="utf-8"
   Content-Disposition:attachment;filename=agenda.html
   Content-Length: 59
   Prefer: return=representation

   <html>
     <body>
       <h1>Agenda</h1>
     </body>
   </html>

```
>> Response <<

HTTP/1.1 201 Created
Content-Type: text/calendar; charset="utf-8"
Content-Length: 371
Content-Location: https://cal.example.com/events/64.ics
ETag: "123456789-000-111"
Cal-Managed-ID: 97S

BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//Example Corp.//CalDAV Server//EN
BEGIN:VEVENT
UID:20010712T182145Z-123401@example.com
DTSTAMP:20120201T203412Z
DTSTART:20120714T170000Z
DTEND:20120715T040000Z
SUMMARY:One-off meeting
ATTACH;MANAGED-ID=97S;FMTTYPE=text/html;SIZE=59;
 FILENAME=agenda.html:https://cal.example.com/attach/64/34X22R
END:VEVENT
END:VCALENDAR
```

3.5. Updating Attachments

When an attachment is updated, the server MUST change the associated
"MANAGED-ID" property parameter and MAY change the "ATTACH" property
value. With this approach, clients are able to determine when an
attachment has been updated by some other client by looking for a
change to either the "ATTACH" property value or the "MANAGED-ID"
property parameter value.

To change the data of an existing managed attachment in a calendar
object resource, the following needs to occur:

1. The client issues a POST request targeted at the calendar object
    resource structured as follows:

    A. The Request-URI will include an "action" query parameter with
        the value "attachment-update" (see Section 3.3.1).

    B. The Request-URI will include a "managed-id" query parameter
        with the value matching that of the "MANAGED-ID" property
        parameter for the "ATTACH" property being updated (see
        Section 3.3.3).

    C. The body of the request contains the updated data for the
        attachment.

D. The client MUST include a valid Content-Type header field
    describing the media type of the attachment (as required by
    HTTP).

E. The client SHOULD include a Content-Disposition header field
    [RFC6266] with a "type" parameter set to "attachment", and a
    "filename" parameter that indicates the name of the
    attachment.

F. The client MAY include a Prefer header field [RFC7240] with
    the "return=representation" preference to request that the
    modified calendar object resource be returned as the body of
    a successful response to the POST request.

2. When the server receives the POST request, it does the following:

A. Validates that the "managed-id" query parameter is valid for
    the calendar object resource.

B. Updates the content of the attachment resource corresponding
    to that "managed-id" value with the supplied attachment data.

C. For each affected recurrence instance in the calendar object
    resource targeted by the request, updates the "ATTACH"
    property whose "MANAGED-ID" property parameter value matches
    the "managed-id" query parameter. The "MANAGED-ID" property
    parameter value is changed to allow other clients to detect
    the update, and the property value (attachment URI) might
    also be changed. The "ATTACH" property SHOULD contain a
    "FMTTYPE" property parameter whose value matches the Content-
    Type header field value from the request; this could differ
    from the original value if the media type of the updated
    attachment is different. The "ATTACH" property SHOULD
    contain a "FILENAME" property parameter whose value matches
    the Content-Disposition header field "filename" parameter
    value from the request, taking into account the restrictions
    expressed in Section 4.2. The "ATTACH" property SHOULD
    include a "SIZE" property parameter whose value represents
    the size in octets of the updated attachment.

D. Upon successful update of the attachment resource, and
    modification of the targeted calendar object resource, it
    MUST return an appropriate HTTP success status response and
    include a "Cal-Managed-ID" header field containing the new
    value of the "MANAGED-ID" property parameter. The client can
    use the "Cal-Managed-ID" header field value to correlate the
    attachment with "ATTACH" properties added to the calendar

object resource. If the client included a Prefer header
field with the "return=representation" preference in the
request, the server SHOULD return the modified calendar
object resource as the body of the response. Otherwise, the
server can expect that the client will reload the calendar
object resource with a subsequent GET request to refresh any
local cache.

The update operation does not take a "rid" query parameter and does
not add, or remove, any "ATTACH" property in the targeted calendar
object resource. To link an existing attachment to a new instance,
the client simply does a PUT on the calendar object resource, adding
an "ATTACH" property that duplicates the existing one (see
Section 3.7).

In the following example, the client updates an existing attachment
and asks the server (via the Prefer header field [RFC7240]) to return
the updated version of that event in the response.

   >> Request <<

   POST /events/64.ics?action=attachment-update&managed-id=97S HTTP/1.1
   Host: cal.example.com
   Content-Type: text/html; charset="utf-8"
   Content-Disposition:attachment;filename=agenda.html
   Content-Length: 96
   Prefer: return=representation

   <html>
     <body>
       <h1>Agenda</h1>
       <p>Discuss attachment draft</p>
     </body>
   </html>

```
>> Response <<

HTTP/1.1 200 OK
Content-Type: text/calendar; charset="utf-8"
Content-Length: 371
Content-Location: https://cal.example.com/events/64.ics
Cal-Managed-ID: 98S
ETag: "123456789-000-222"

BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//Example Corp.//CalDAV Server//EN
BEGIN:VEVENT
UID:20010712T182145Z-123401@example.com
DTSTAMP:20120201T203412Z
DTSTART:20120714T170000Z
DTEND:20120715T040000Z
SUMMARY:One-off meeting
ATTACH;MANAGED-ID=98S;FMTTYPE=text/html;SIZE=96;
 FILENAME=agenda.html:https://cal.example.com/attach/64/34X22R
END:VEVENT
END:VCALENDAR
```

3.6. Removing Attachments via POST

To remove an existing attachment from a calendar object, the
following needs to occur:

1. The client issues a POST request targeted at the calendar object
    resource structured as follows:

    A. The Request-URI will include an "action" query parameter with
        the value "attachment-remove" (see Section 3.3.1).

    B. If all recurrence instances are having an attachment removed,
        the "rid" query parameter is not present in the Request-URI.
        If one or more specific recurrence instances are targeted,
        then the Request-URI will include a "rid" query parameter
        containing the list of instances (see Section 3.3.2).

    C. The Request-URI will include a "managed-id" query parameter
        with the value matching that of the "MANAGED-ID" property
        parameter for the "ATTACH" property being removed (see
        Section 3.3.3).

    D. The body of the request will be empty.

   E. The client MAY include a Prefer header field [RFC7240] with
       the "return=representation" preference to request that the
       modified calendar object resource be returned as the body of
       a successful response to the POST request.

2. When the server receives the POST request, it does the following:

   A. Validates that any recurrence instances referred to via the
       "rid" query parameter are valid for the calendar object
       resource being targeted.

   B. Validates that the "managed-id" query parameter is valid for
       the calendar object resource and specific instances being
       targeted.

   C. For each affected recurrence instance in the calendar object
       resource targeted by the request, removes the matching
       "ATTACH" property. Note that if a specified recurrence
       instance does not have a matching component in the calendar
       object resource, then the server MUST modify the calendar
       object resource to include an overridden component with the
       appropriate "RECURRENCE-ID" property and the matching
       "ATTACH" property removed. This latter case is actually
       valid only if the master component does include the
       referenced "ATTACH" property.

   D. If the attachment resource is no longer referenced by any
       instance of the calendar object resource, it can delete the
       attachment resource to free up storage space.

   E. Upon successful removal of the attachment resource and
       modification of the targeted calendar object resource, it
       MUST return an appropriate HTTP success status response. If
       the client included a Prefer header field with the
       "return=representation" preference in the request, the server
       SHOULD return the modified calendar object resource as the
       body of the response. Otherwise, the server can expect that
       the client will reload the calendar object resource with a
       subsequent GET request to refresh any local cache.

In the following example, the client deletes an existing attachment
by passing its "managed-id" value in the request. The Prefer header
field [RFC7240] is not set in the request so the calendar object
resource data is not returned in the response.

```
   >> Request <<

   POST /events/64.ics?action=attachment-remove&managed-id=98S HTTP/1.1
   Host: cal.example.com
   Content-Length: 0

   >> Response <<

   HTTP/1.1 204 No Content
   Content-Length: 0
```

3.7. Adding Existing Managed Attachments via PUT

   Clients can make use of existing managed attachments by adding the
   corresponding "ATTACH" property to calendar object resources (subject
   to the restrictions described in Section 3.12.2).

   If a managed attachment is used in more than calendar resource,
   servers SHOULD NOT change either the "MANAGED-ID" property parameter
   value or the "ATTACH" property value for these attachments; this
   ensures that clients do not have to download the attachment data
   again if they already have it cached. Additionally, servers SHOULD
   validate "SIZE" property parameter values and replace incorrect
   values with the actual sizes of existing attachments.

   These PUT requests are subject to the preconditions listed in
   Section 3.11.

3.8. Updating Attachments via PUT

   Servers MUST NOT allow clients to update attachment data directly via
   a PUT on the attachment URI (or via any other HTTP method that
   modifies content). Instead, attachments can only be updated via use
   of POST requests on the calendar data.

3.9. Removing Attachments via PUT

   Clients can remove attachments by simply rewriting the calendar
   object resource data to remove the appropriate "ATTACH" properties.
   Servers MUST NOT allow clients to delete attachments directly via a
   DELETE request on the attachment URI.

3.10. Retrieving Attachments

   Clients retrieve attachments by issuing an HTTP GET request using the
   value of the corresponding "ATTACH" property as the Request-URI,
   taking into account the substitution mechanism associated with the
   "CALDAV:managed-attachments-server-URL" property (see Section 6.1).

3.11. Error Handling

   This specification creates additional preconditions for the POST
   method.

   The new preconditions are:

   (CALDAV:max-attachment-size): The attachment submitted in the POST
      request MUST have an octet size less than or equal to the value of
      the "CALDAV:max-attachment-size" property value (Section 6.2) on
      the calendar collection of the target calendar resource.

   (CALDAV:max-attachments-per-resource): The addition of the
      attachment submitted in the POST request MUST result in the target
      calendar resource having a number of managed attachments less than
      or equal to the value of the "CALDAV:max-attachments-per-resource"
      property value (Section 6.3) on the calendar collection of the
      target calendar resource.

   (CALDAV:valid-action): The "action" query parameter in the POST
      request MUST contain only one of the following three values:
      "attachment-add", "attachment-update", or "attachment-remove".

   (CALDAV:valid-rid): The "rid" query parameter in the POST request
      MUST NOT be present with an "action=attachment-update" query
      parameter and MUST contain the value "M" and/or values
      corresponding to "RECURRENCE-ID" property values in the iCalendar
      data targeted by the request.

   (CALDAV:valid-managed-id): The "managed-id" query parameter in the
      POST request MUST NOT be present with an "action=attachment-add"
      query parameter and MUST contain a value corresponding to a
      "MANAGED-ID" property parameter value in the iCalendar data
      targeted by the request.

   A POST request to add, modify, or delete a managed attachment results
   in an implicit modification of the targeted calendar resource
   (equivalent of a PUT). As a consequence, clients should also be
   prepared to handle preconditions associated with this implicit PUT.
   This includes (but is not limited to):

      (CALDAV:max-resource-size) (from Section 5.3.2.1 of [RFC4791])

      (DAV:quota-not-exceeded) (from Section 6 of [RFC4331])

      (DAV:sufficient-disk-space) (from Section 6 of [RFC4331])

   A PUT request to add or modify an existing calendar object resource
   can make reference to an existing managed attachment. The following
   new precondition is defined:

   (CALDAV:valid-managed-id-parameter): a "MANAGED-ID" property
      parameter value in the iCalendar data in the PUT request is not
      valid (e.g., does not match any existing managed attachment).

   If a precondition for a request is not satisfied:

   1. The response status of the request MUST either be 403 (Forbidden)
       if the request should not be repeated because it will always
       fail, or 409 (Conflict) if it is expected that the user might be
       able to resolve the conflict and resubmit the request.

   2. The appropriate XML element MUST be returned as the child of a
       top-level DAV:error element in the response body.

3.12. Additional Considerations

3.12.1. Quotas

   The WebDAV Quotas specification [RFC4331] defines two live WebDAV
   properties (DAV:quota-available-bytes and DAV:quota-used-bytes) to
   communicate storage quota information to clients. Server
   implementations MAY choose to include managed attachment sizes when
   calculating the amount of storage used by a particular resource.

3.12.2. Access Control

   Access to the managed attachments referenced in a calendar object
   resource SHOULD be restricted to only those calendar users who have
   access to that calendar object either directly or indirectly (via
   being an attendee who would receive a scheduling message).

   When accessing a managed attachment, clients SHOULD be prepared to
   authenticate with the server storing the attachment resource. The
   credentials required to access the managed attachment store could be
   different from the ones used to access the CalDAV server.

   This specification only allows organizers of scheduled events to add
   managed attachments. Servers MUST prevent attendees of scheduled
   events from adding, updating, or removing managed attachments. In
   addition, the server MUST prevent a calendar user from reusing a
   managed attachment (based on its "managed-id" value), unless that
   user is the one who originally created the managed attachment.

3.12.3. Redirects

   For POST requests that add or update attachment data, the server MAY
   issue a 307 (Temporary Redirect) [RFC7231] or 308 (Permanent
   Redirect) [RFC7538] response to require the client to reissue the
   POST request using a different Request-URI. As a result, clients
   SHOULD use the "100-continue" expectation defined in Section 5.1.1 of
   [RFC7231]. Using this mechanism ensures that, if a redirect does
   occur, the client does not needlessly send the attachment data.

3.12.4. Processing Time

   Clients can expect servers to take a while to respond to POST
   requests that include large attachment bodies. Servers SHOULD use
   the 102 (Processing) interim response defined in Section 10.1 of
   [RFC2518] to keep the client connection alive if the POST request
   will take significant time to complete.

3.12.5. Automatic Cleanup by Servers

   Servers MAY automatically remove attachment data, for example, to
   regain the storage taken by unused attachments or as the result of a
   virus scanning. When doing so, they SHOULD NOT modify calendar data
   referencing those attachments. Instead, they SHOULD respond with 410
   (Gone) to any request on the removed attachment URI.

3.12.6. Sending Scheduling Messages with Attachments

   When a managed attachment is added, updated, or removed from a
   calendar object resource, the server MUST ensure that a scheduling
   message is sent to update any attendees with the changes, as per
   [RFC6638].

3.12.7. Migrating Calendar Data

   When exporting calendar data from a CalDAV server supporting managed
   attachments, clients SHOULD remove all "MANAGED-ID" property
   parameters from "ATTACH" properties in the calendar data. Similarly,
   when importing calendar data from another source, clients SHOULD
   remove any "MANAGED-ID" property parameters on "ATTACH" properties
   (failure to do so will likely result in the server removing those
   properties automatically).

4. Modifications to iCalendar Syntax

4.1. SIZE Property Parameter

   Parameter Name: SIZE

   Purpose: To specify the size of an attachment.

   Format Definition: This property parameter is defined by the
      following notation:

      sizeparam = "SIZE" "=" paramtext
      ; positive integers

   Description: This property parameter MAY be specified on "ATTACH"
      properties. It indicates the size in octets of the corresponding
      attachment data. Since iCalendar integer values are restricted to
      a maximum value of 2147483647, the current property parameter is
      defined as text to allow an extended range to be used.

   Example:

      ATTACH;SIZE=1234:https://attachments.example.com/abcd.txt

4.2. FILENAME Property Parameter

   Parameter Name: FILENAME

   Purpose: To specify the filename of a managed attachment.

   Format Definition: This property parameter is defined by the
      following notation:

      filenameparam = "FILENAME" "=" paramtext

   Description: This property parameter MAY be specified on "ATTACH"
      properties corresponding to managed attachments. Its value
      provides information on how to construct a filename for storing
      the attachment data. This parameter is very similar in nature to
      the Content-Disposition header field "filename" parameter and
      exposes the same security risks. As a consequence, clients MUST
      follow the guidelines expressed in Section 4.3 of [RFC6266] when
      consuming this property parameter value. Similarly, servers MUST
      follow those same guidelines before storing a value.

   Example:

      ATTACH;FILENAME=agenda.html:
       https://attachments.example.com/rt452S

4.3. MANAGED-ID Property Parameter

   Parameter Name: MANAGED-ID

   Purpose: To uniquely identify a managed attachment.

   Format Definition: This property parameter is defined by the
      following notation:

      managedidparam = "MANAGED-ID" "=" paramtext

   Description: This property parameter MUST be specified on "ATTACH"
      properties corresponding to managed attachments. Its value is
      generated by the server and uniquely identifies a managed
      attachment within the scope of the CalDAV server. This property
      parameter MUST NOT be present in the case of unmanaged
      attachments.

   Example:

      ATTACH;MANAGED-ID=aUNhbGVuZGFy:
       https://attachments.example.com/abcd.txt

5. Additional Message Header Fields

5.1. Cal-Managed-ID Response Header Field

   The Cal-Managed-ID response header field provides the value of the
   "MANAGED-ID" property parameter corresponding to a newly added
   "ATTACH" property.

   ABNF:

   Cal-Managed-ID = "Cal-Managed-ID" ":" paramtext
   ; "paramtext" is defined in Section 3.1 of [RFC5545]

   Example:

   Cal-Managed-ID:aUNhbGVuZGFy

The Cal-Managed-ID header field MUST only be sent by an origin server
in response to a successful POST request with an "action" query
parameter set to "attachment-add" or "attachment-update". It MUST
only appear once in a response and MUST NOT appear in trailers.

The Cal-Managed-ID header field is end to end and MUST be forwarded
by intermediaries. Intermediaries MUST NOT insert, delete, or modify
a Cal-Managed-ID header field.

6. Additional WebDAV Properties

6.1. CALDAV:managed-attachments-server-URL Property

   Name: managed-attachments-server-URL

   Namespace: urn:ietf:params:xml:ns:caldav

   Purpose: This property specifies the server base URI to use when
      retrieving managed attachments.

   Protected: This property MUST be protected as only the server can
      update the value.

   COPY/MOVE behavior: This property is only defined on a calendar home
      collection, which cannot be moved or copied.

   allprop behavior: This property SHOULD NOT be returned by a PROPFIND
      DAV:allprop request.

   Description: This property MAY be defined on a calendar home
      collection. If present, it contains either a single DAV:href XML
      element or none at all.

      When one DAV:href element is present, its value MUST be an
      absolute HTTP URI containing only the scheme (i.e., "https") and
      authority (i.e., host and port) parts. Whenever a managed
      attachment is to be retrieved via an HTTP GET, the client MUST
      construct the actual URL of the attachment by substituting the
      scheme and authority parts of the attachment URI (as stored in the
      iCalendar "ATTACH" property) with the present WebDAV property
      value.

      When no DAV:href element is present, the client MUST substitute
      the scheme and authority parts of the attachment URI with the
      scheme and authority part of the calendar home collection absolute
      URI.

   In the absence of this property, the client can consider the
   attachment URI as its actual URL.

Definition:

<!ELEMENT managed-attachments-server-URL (DAV:href?)>

Example:

<C:managed-attachments-server-URL xmlns:D="DAV:"
   xmlns:C="urn:ietf:params:xml:ns:caldav">
  <D:href>https://attachstore.example.com</D:href>
</C:managed-attachments-server-URL>

6.2. CALDAV:max-attachment-size Property

Name: max-attachment-size

Namespace: urn:ietf:params:xml:ns:caldav

Purpose: This property provides a numeric value indicating the
   maximum attachment size, in octets, that the server is willing to
   accept when a managed attachment is stored on the server.

Protected: This property MUST be protected as it indicates limits
   provided by the server.

COPY/MOVE behavior: This property value MUST be preserved in COPY
   and MOVE operations.

allprop behavior: This property SHOULD NOT be returned by a PROPFIND
   DAV:allprop request.

Description: The "CALDAV:max-attachment-size" property is used to
   specify a numeric value that represents the maximum attachment
   size, in octets, that the server is willing to accept when a
   managed attachment is stored on the server. The property is
   defined on the parent collection of the calendar object resource
   to which the attachment is associated. Any attempt to store a
   managed attachment exceeding this size MUST result in an error,
   with the CALDAV:max-attachment-size precondition (Section 3.11)
   being violated. In the absence of this property, the client can
   assume that the server will allow storing an attachment of any
   reasonable size.

   Definition:

   <!ELEMENT max-attachment-size (#PCDATA)>
   <!-- PCDATA value: a numeric value (positive decimal integer) -->

   Example:

   <C:max-attachment-size xmlns:C="urn:ietf:params:xml:ns:caldav"
        >102400000</C:max-attachment-size>

6.3. CALDAV:max-attachments-per-resource Property

   Name: max-attachments-per-resource

   Namespace: urn:ietf:params:xml:ns:caldav

   Purpose: This property provides a numeric value indicating the
      maximum number of managed attachments across all instances of a
      calendar object resource stored in a calendar collection.

   Protected: This property MUST be protected as it indicates limits
      provided by the server.

   COPY/MOVE behavior: This property value MUST be preserved in COPY
      and MOVE operations.

   allprop behavior: This property SHOULD NOT be returned by a PROPFIND
      DAV:allprop request.

   Description: The "CALDAV:max-attachments-per-resource" property is
      used to specify a numeric value that represents the maximum number
      of managed attachments across all instances of a calendar object
      resource stored in a calendar collection. Unmanaged attachments
      are not counted toward that limit. The property is defined on the
      parent collection of the calendar object resource to which the
      attachment is associated. Any attempt to add a managed attachment
      that would cause the calendar resource to exceed this limit MUST
      result in an error, with the CALDAV:max-attachments-per-resource
      precondition (Section 3.11) being violated. In the absence of
      this property, the client can assume that the server can handle
      any number of managed attachments per calendar resource.

   Definition:

   <!ELEMENT max-attachments-per-resource (#PCDATA)>
   <!-- PCDATA value: a numeric value (positive decimal integer) -->

   Example:

   <C:max-attachments-per-resource
      xmlns:C="urn:ietf:params:xml:ns:caldav"
       >12</C:max-attachments-per-resource>

7. Security Considerations

   The security considerations in [RFC4791] and [RFC4918] apply to this
   extension. Additionally, servers need to be aware that a client
   could attack underlying storage by POSTing extremely large
   attachments and could attack processing time by uploading a recurring
   event with a large number of overrides and then repeatedly adding,
   updating, and deleting attachments.

   Malicious content could be introduced into the calendar server by way
   of a managed attachment, and propagated to many end users via
   scheduling. Servers SHOULD check managed attachments for malicious
   or inappropriate content. Upon detecting such content, servers
   SHOULD remove the attachment following the rules described in
   Section 3.12.5.

8. IANA Considerations

8.1. Parameter Registrations

   This specification defines the following new iCalendar property
   parameters to be added to the "Parameters" registry defined in
   Section 8.2.4 of [RFC5545]:

            +------------+---------+-----------------------+
            | Parameter | Status | Reference |
            +------------+---------+-----------------------+
            | SIZE | Current | RFC 8607, Section 4.1 |
            | FILENAME | Current | RFC 8607, Section 4.2 |
            | MANAGED-ID | Current | RFC 8607, Section 4.3 |
            +------------+---------+-----------------------+

8.2. Message Header Field Registrations

   The message header fields below should be added to the "Permanent
   Message Header Field Names" registry (see [RFC3864]).

8.2.1. Cal-Managed-ID

   Header field name: Cal-Managed-ID

   Protocol: http

   Status: standard

   Author/Change controller: IETF

   Reference: this specification (Section 5.1)

   Related information: none

9. References

9.1. Normative References

   [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
              Requirement Levels", BCP 14, RFC 2119,
              DOI 10.17487/RFC2119, March 1997,
              <https://www.rfc-editor.org/info/rfc2119>.

   [RFC2518] Goland, Y., Whitehead, E., Faizi, A., Carter, S., and D.
              Jensen, "HTTP Extensions for Distributed Authoring --
              WEBDAV", RFC 2518, DOI 10.17487/RFC2518, February 1999,
              <https://www.rfc-editor.org/info/rfc2518>.

   [RFC3864] Klyne, G., Nottingham, M., and J. Mogul, "Registration
              Procedures for Message Header Fields", BCP 90, RFC 3864,
              DOI 10.17487/RFC3864, September 2004,
              <https://www.rfc-editor.org/info/rfc3864>.

   [RFC4331] Korver, B. and L. Dusseault, "Quota and Size Properties
              for Distributed Authoring and Versioning (DAV)
              Collections", RFC 4331, DOI 10.17487/RFC4331, February
              2006, <https://www.rfc-editor.org/info/rfc4331>.

   [RFC4791] Daboo, C., Desruisseaux, B., and L. Dusseault,
              "Calendaring Extensions to WebDAV (CalDAV)", RFC 4791,
              DOI 10.17487/RFC4791, March 2007,
              <https://www.rfc-editor.org/info/rfc4791>.

   [RFC4918] Dusseault, L., Ed., "HTTP Extensions for Web Distributed
              Authoring and Versioning (WebDAV)", RFC 4918,
              DOI 10.17487/RFC4918, June 2007,
              <https://www.rfc-editor.org/info/rfc4918>.

   [RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax
              Specifications: ABNF", STD 68, RFC 5234,
              DOI 10.17487/RFC5234, January 2008,
              <https://www.rfc-editor.org/info/rfc5234>.

   [RFC5545] Desruisseaux, B., Ed., "Internet Calendaring and
              Scheduling Core Object Specification (iCalendar)",
              RFC 5545, DOI 10.17487/RFC5545, September 2009,
              <https://www.rfc-editor.org/info/rfc5545>.

   [RFC6266] Reschke, J., "Use of the Content-Disposition Header Field
              in the Hypertext Transfer Protocol (HTTP)", RFC 6266,
              DOI 10.17487/RFC6266, June 2011,
              <https://www.rfc-editor.org/info/rfc6266>.

   [RFC6638] Daboo, C. and B. Desruisseaux, "Scheduling Extensions to
              CalDAV", RFC 6638, DOI 10.17487/RFC6638, June 2012,
              <https://www.rfc-editor.org/info/rfc6638>.

   [RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer
              Protocol (HTTP/1.1): Message Syntax and Routing",
              RFC 7230, DOI 10.17487/RFC7230, June 2014,
              <https://www.rfc-editor.org/info/rfc7230>.

   [RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer
              Protocol (HTTP/1.1): Semantics and Content", RFC 7231,
              DOI 10.17487/RFC7231, June 2014,
              <https://www.rfc-editor.org/info/rfc7231>.

   [RFC7240] Snell, J., "Prefer Header for HTTP", RFC 7240,
              DOI 10.17487/RFC7240, June 2014,
              <https://www.rfc-editor.org/info/rfc7240>.

   [RFC7538] Reschke, J., "The Hypertext Transfer Protocol Status Code
              308 (Permanent Redirect)", RFC 7538, DOI 10.17487/RFC7538,
              April 2015, <https://www.rfc-editor.org/info/rfc7538>.

   [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC
              2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,
              May 2017, <https://www.rfc-editor.org/info/rfc8174>.

9.2. Informative References

   [RFC5023] Gregorio, J., Ed. and B. de hOra, Ed., "The Atom
              Publishing Protocol", RFC 5023, DOI 10.17487/RFC5023,
              October 2007, <https://www.rfc-editor.org/info/rfc5023>.

   [RFC5546] Daboo, C., Ed., "iCalendar Transport-Independent
              Interoperability Protocol (iTIP)", RFC 5546,
              DOI 10.17487/RFC5546, December 2009,
              <https://www.rfc-editor.org/info/rfc5546>.

   [RFC7320] Nottingham, M., "URI Design and Ownership", BCP 190,
              RFC 7320, DOI 10.17487/RFC7320, July 2014,
              <https://www.rfc-editor.org/info/rfc7320>.

   [RFC8144] Murchison, K., "Use of the Prefer Header Field in Web
              Distributed Authoring and Versioning (WebDAV)", RFC 8144,
              DOI 10.17487/RFC8144, April 2017,
              <https://www.rfc-editor.org/info/rfc8144>.

Appendix A. Example Involving Recurring Events

   In the following example, the organizer of a recurring meeting makes
   an unsuccessful attempt to add an agenda (HTML attachment) to the
   corresponding calendar resource with a conditional request. Note
   that the client includes both the Expect and Prefer header fields in
   the request, thereby preventing itself from needlessly sending the
   attachment data and requesting that the current resource be returned
   in the failure response (see Section 3.2 of [RFC8144]).

   >> Request <<

   POST /events/65.ics?action=attachment-add HTTP/1.1
   Host: cal.example.com
   Content-Type: text/html; charset="utf-8"
   Content-Disposition: attachment;filename=agenda.html
   Content-Length: 80
   If-Match: "abcdefg-000"
   Expect: 100-continue
   Prefer: return=representation

```
   >> Final Response <<

   HTTP/1.1 412 Precondition Failed
   Content-Type: text/calendar; charset="utf-8"
   Content-Length: 929
   Content-Location: https://cal.example.com/events/65.ics
   ETag: "123456789-000-000"


   BEGIN:VCALENDAR
   VERSION:2.0
   PRODID:-//Example Corp.//CalDAV Server//EN
   BEGIN:VTIMEZONE
   LAST-MODIFIED:20040110T032845Z
   TZID:America/Montreal
   BEGIN:DAYLIGHT
   DTSTART:20000404T020000
   RRULE:FREQ=YEARLY;BYDAY=1SU;BYMONTH=4
   TZNAME:EDT
   TZOFFSETFROM:-0500
   TZOFFSETTO:-0400
   END:DAYLIGHT
   BEGIN:STANDARD
   DTSTART:20001026T020000
   RRULE:FREQ=YEARLY;BYDAY=-1SU;BYMONTH=10
   TZNAME:EST
   TZOFFSETFROM:-0400
   TZOFFSETTO:-0500
   END:STANDARD
   END:VTIMEZONE
   BEGIN:VEVENT
   UID:20010712T182145Z-123401@example.com
   DTSTAMP:20120201T203412Z
   DTSTART;TZID=America/Montreal:20120206T100000
   DURATION:PT1H
   RRULE:FREQ=WEEKLY
   SUMMARY:Planning Meeting
   ORGANIZER:mailto:cyrus@example.com
   ATTENDEE;CUTYPE=INDIVIDUAL;PARTSTAT=ACCEPTED:mailto:cyrus@exampl
    e.com
   ATTENDEE;CUTYPE=INDIVIDUAL;PARTSTAT=ACCEPTED:mailto:arnaudq@exam
    ple.com
   ATTENDEE;CUTYPE=INDIVIDUAL;PARTSTAT=NEEDS-ACTION:mailto:mike@exa
    mple.com
   END:VEVENT
   END:VCALENDAR
```

   The organizer of a recurring meeting successfully adds an agenda
   (HTML attachment) to the corresponding calendar resource. Attendees
   of the meeting are granted read access to the newly created
   attachment resource. Their own copy of the meeting is updated to
   include the new "ATTACH" property pointing to the attachment
   resource, and they are notified of the change via their scheduling
   inbox.

   >> Request <<

   POST /events/65.ics?action=attachment-add HTTP/1.1
   Host: cal.example.com
   Content-Type: text/html; charset="utf-8"
   Content-Disposition: attachment;filename=agenda.html
   Content-Length: 80
   If-Match: "123456789-000-000"
   Expect: 100-continue
   Prefer: return=representation

   >> Interim Response <<

   HTTP/1.1 100 Continue

   >> Request Body <<

   <html>
     <body>
       <h1>Agenda</h1>
       <p>As usual</p>
     </body>
   </html>

```
   >> Final Response <<

   HTTP/1.1 201 Created
   Content-Type: text/calendar; charset="utf-8"
   Content-Length: 1043
   Content-Location: https://cal.example.com/events/65.ics
   ETag: "123456789-000-111"
   Cal-Managed-ID: 97S


   BEGIN:VCALENDAR
   VERSION:2.0
   PRODID:-//Example Corp.//CalDAV Server//EN
   BEGIN:VTIMEZONE
   LAST-MODIFIED:20040110T032845Z
   TZID:America/Montreal
   BEGIN:DAYLIGHT
   DTSTART:20000404T020000
   RRULE:FREQ=YEARLY;BYDAY=1SU;BYMONTH=4
   TZNAME:EDT
   TZOFFSETFROM:-0500
   TZOFFSETTO:-0400
   END:DAYLIGHT
   BEGIN:STANDARD
   DTSTART:20001026T020000
   RRULE:FREQ=YEARLY;BYDAY=-1SU;BYMONTH=10
   TZNAME:EST
   TZOFFSETFROM:-0400
   TZOFFSETTO:-0500
   END:STANDARD
   END:VTIMEZONE
   BEGIN:VEVENT
   UID:20010712T182145Z-123401@example.com
   DTSTAMP:20120201T203412Z
   DTSTART;TZID=America/Montreal:20120206T100000
   DURATION:PT1H
   RRULE:FREQ=WEEKLY
   SUMMARY:Planning Meeting
   ORGANIZER:mailto:cyrus@example.com
   ATTENDEE;CUTYPE=INDIVIDUAL;PARTSTAT=ACCEPTED:mailto:cyrus@exampl
    e.com
   ATTENDEE;CUTYPE=INDIVIDUAL;PARTSTAT=ACCEPTED:mailto:arnaudq@exam
    ple.com
   ATTENDEE;CUTYPE=INDIVIDUAL;PARTSTAT=NEEDS-ACTION:mailto:mike@exa
    mple.com
   ATTACH;MANAGED-ID=97S;FMTTYPE=text/html;SIZE=80;
    FILENAME=agenda.html:https://cal.example.com/attach/65/34X22R
   END:VEVENT
   END:VCALENDAR
```

   The organizer has a more specific agenda for the 20th of February
   meeting. It is added to that particular instance of the meeting by
   specifying the "rid" query parameter. Note that an overridden
   instance is created with the "RECURRENCE-ID" property value matching
   the value of the "rid" query parameter in the request. Also, note
   that the server takes significant time to complete the request and
   notifies the client accordingly.

   >> Request <<

POST /events/65.ics?action=attachment-add&rid=20120220T100000 HTTP/1.1
Host: cal.example.com
Content-Type: text/html; charset="utf-8"
Content-Disposition: attachment;filename=agenda0220.html
Content-Length: 105
If-Match: "123456789-000-111"
Expect: 100-continue
Prefer: return=representation

   >> Interim Response <<

   HTTP/1.1 100 Continue

   >> Request Body <<

   <html>
     <body>
       <h1>Agenda</h1>
       <p>Something different, for a change</p>
     </body>
   </html>

   >> Interim Response <<

   HTTP/1.1 102 Processing

   >> Final Response <<

   HTTP/1.1 201 Created
   Content-Type: text/calendar; charset="utf-8"
   Content-Length: 1661
   Content-Location: https://cal.example.com/events/65.ics
   ETag: "123456789-000-222"
   Cal-Managed-ID: 33225

```
BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//Example Corp.//CalDAV Server//EN
BEGIN:VTIMEZONE
LAST-MODIFIED:20040110T032845Z
TZID:America/Montreal
BEGIN:DAYLIGHT
DTSTART:20000404T020000
RRULE:FREQ=YEARLY;BYDAY=1SU;BYMONTH=4
TZNAME:EDT
TZOFFSETFROM:-0500
TZOFFSETTO:-0400
END:DAYLIGHT
BEGIN:STANDARD
DTSTART:20001026T020000
RRULE:FREQ=YEARLY;BYDAY=-1SU;BYMONTH=10
TZNAME:EST
TZOFFSETFROM:-0400
TZOFFSETTO:-0500
END:STANDARD
END:VTIMEZONE
BEGIN:VEVENT
UID:20010712T182145Z-123401@example.com
DTSTAMP:20120201T203412Z
DTSTART;TZID=America/Montreal:20120206T100000
DURATION:PT1H
RRULE:FREQ=WEEKLY
SUMMARY:Planning Meeting
ORGANIZER:mailto:cyrus@example.com
ATTENDEE;CUTYPE=INDIVIDUAL;PARTSTAT=ACCEPTED:mailto:cyrus@exampl
 e.com
ATTENDEE;CUTYPE=INDIVIDUAL;PARTSTAT=ACCEPTED:mailto:arnaudq@exam
 ple.com
ATTENDEE;CUTYPE=INDIVIDUAL;PARTSTAT=NEEDS-ACTION:mailto:mike@exa
 mple.com
ATTACH;MANAGED-ID=97S;FMTTYPE=text/html;SIZE=80;
 FILENAME=agenda.html:https://cal.example.com/attach/65/34X22R
END:VEVENT
BEGIN:VEVENT
UID:20010712T182145Z-123401@example.com
RECURRENCE-ID;TZID=America/Montreal:20120220T100000
DTSTAMP:20120201T203412Z
DTSTART;TZID=America/Montreal:20120220T100000
DURATION:PT1H
SUMMARY:Planning Meeting
ORGANIZER:mailto:cyrus@example.com
ATTENDEE;CUTYPE=INDIVIDUAL;PARTSTAT=ACCEPTED:mailto:cyrus@example.
 com
```

```
   ATTENDEE;CUTYPE=INDIVIDUAL;PARTSTAT=ACCEPTED:mailto:arnaudq@exampl
    e.com
   ATTENDEE;CUTYPE=INDIVIDUAL;PARTSTAT=NEEDS-ACTION:mailto:mike@examp
    le.com
   ATTACH;MANAGED-ID=33225;FMTTYPE=text/html;SIZE=105;
    FILENAME=agenda0220.html:https://cal.example.com/attach/65/FGZ225
   END:VEVENT
   END:VCALENDAR
```

Acknowledgments

   This specification came about via discussions at the Calendaring and
   Scheduling Consortium. Thanks in particular to Mike Douglass and
   Eric York.

Authors' Addresses

   Cyrus Daboo
   Apple Inc.
   1 Infinite Loop
   Cupertino, CA 95014
   United States of America

   Email: cyrus@daboo.name
   URI: http://www.apple.com/

   Arnaud Quillaud
   Oracle Corporation
   180, Avenue de l'Europe
   Saint Ismier cedex 38334
   France

   Email: arnaud.quillaud@oracle.com
   URI: http://www.oracle.com/

   Kenneth Murchison (editor)
   FastMail US LLC
   1429 Walnut St, Suite 1201
   Philadephia, PA 19102
   United States of America

   Email: murch@fastmailteam.com
   URI: http://www.fastmail.com/