CALCONNECT THE CALENDARING & SCHEDULING CONSORTIUM

CALCONNECT DOCUMENT CD1011

Type:	**Proposal**
Title:	**CalWS-Rest - Restful Web Services Protocol for Calendaring**
Version:	**1.0.1**
Date:	**2012-02-22**
Status:	**Published**
Source:	**TC XML**

This document was editorially updated to ensure synchronization with the OASIS WS-Calendar publication of CalWS-SOAP as V1.0 for public review and contains the same content..

CalWSRest - Restful Web service protocol for calendaring

Version 1.0.1

Working Draft 05

17 February 2012

Technical Committee:
CalConnect TC-XML

Chair:
Michael Douglass (douglm@rpi.edu) Rensselaer Polytechnic Institute

Editor:
Michael Douglass (douglm@rpi.edu) Rensselaer Polytechnic Institute

Related work:
This specification is related to:

- *WS-Calendar Version 1.0*. Latest version.
  http://docs.oasis-open.org/ws-calendar/ws-calendar/v1.0/ws-calendar-1.0-spec.html

Abstract:
Summary of the technical purpose of the document.

Table of Contents

1 Introduction

The CalWS REST protocol is built upon and makes the same assumptions about structure as the CalDAV protocol defined in **[RFC 4791]** and related specifications. It does NOT require nor assume the WebDAV nor CalDAV protocol.

Calendar resources, for example events and tasks are stored as named resources (files) inside special collections (folders) known as "**Calendar Collections**".

This specification can be looked upon as a layer built on top of CalDAV and defines the basic operations which allow creation, retrieval, update and deletion. In addition, query and freebusy operations are defined to allow efficient, partial retrieval of calendar data.

This does not mean that a CalWS service must be built on CalDAV, merely that a degree of conformity is established such that services built in that manner do not have a significant mismatch. It is assumed that some CalWS REST services will be built without any CalDAV support.

1.1 Terminology

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in **[RFC2119]**.

1.2 Normative References

[RFC2119] S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*, http://www.ietf.org/rfc/rfc2119.txt, IETF RFC 2119, March 1997.

[RFC4791] C. Daboo, B. Desruisseaux, L. Dusseault, *Calendaring Extensions to WebDAV (CalDAV)*, http://www.ietf.org/RFC/RFC4791.txt, IETF RFC4791, March 1997.

[WS-Calendar] *WS-Calendar Version 1.0*. 19 January 2011. OASIS Committee Specification http://docs.oasis-open.org/ws-calendar/ws-calendar-spec/v1.0/cs01/ws-calendar-spec-v1.0-cs01.pdf.

[XRD] Extensible Resource Descriptor (XRD) Version 1.0, 1 November 2010, OASIS Standard, http://docs.oasis-open.org/xri/xrd/v1.0/os/xrd-1.0-os.xml

1.3 Non-Normative References

REST T Fielding, Architectural Styles and the Design of Network-based Software Architectures, http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm.

1.4 Namespace

XML namespaces and prefixes used in this standard:

Table 1 1: XML Namespaces in this standard

Prefix	Namespace
xcal	urn:ietf:params:xml:ns:icalendar-2.0
calWS	http://docs.oasis-open.org/ws-calendar/ns/REST
xrd	http://docs.oasis-open.org/ns/xri/xrd-1.0

35

2 Calendar Services

The Service interactions are built upon and make the same assumptions about structure as the CalDAV protocol defined in **[RFC4791]** and related specifications. It does NOT require nor assume the WebDAV nor CalDAV protocol but does make use of some of the same elements and structures in the CalDAV XML namespace.

Calendar resources, for example events and tasks are stored as named resources (files) inside special collections (folders) known as "**Calendar Collections**".

These services can be looked upon as a layer built on top of CalDAV and defines the basic operations which allow creation, retrieval, update and deletion. In addition, query, and free-busy operations are defined to allow efficient, partial retrieval of calendar data.

These services assume a degree of conformity with CalDAV is established such that services built in that manner do not have a significant mismatch. It is assumed that some WS-Calendar services will be built without any CalDAV support.

2.1 Overview of the protocol

The protocol is an HTTP based RESTfull protocol using a limited set of methods. Each request may be followed by a response containing status information.

The following methods are specified in the protocol description, PUT, POST, GET, DELETE. To avoid various issues with certain methods being blocked clients may use the X-HTTP-Method-Override: header to specify the intended operation. Servers SHOULD behave as if the named method was used.

```
POST /user/fred/calendar/ HTTP/1.1
...
X-HTTP-Method-Override: PUT
Properties
```

A service or resource will have a number of properties which describe the current state of that service or resource. These properties are accessed through a GET on the target resource or service with an ACCEPT header specifying application/xrd+xml. See Section 2.1.3.6

The following operations are defined by this specification:

- Retrieval and update of service and resource properties
- Creation of a calendar object
- Retrieval of a calendar object
- Update of a calendar object
- Deletion of a calendar object
- Query
- Free-busy query

2.1.1 Calendar Object Resources

The same restrictions apply to Calendar Object Resources as specified in CalDAV **[RFC4791]** section 4.2. An additional constraint for CalWS is that no timezone specifications are transferred.

2.1.2 Timezone information

It is assumed that the client and server each have access to a full set of up to date timezone information. Timezones will be referenced by a timezone identifier from the full set of Olson data together with a set of well-known aliases defined **[TZDB]**. CalWS services may advertise themselves as timezone servers through the server properties object.

2.1.3 Issues not addressed by this specification.

A number of issues are not addressed by this version of the specification, either because they should be addressed elsewhere or will be addressed at some later date.

2.1.3.1 Access Control

It is assumed that the targeted server will set an appropriate level of access based on authentication. This specification will not attempt to address the issues of sharing or Access Control Lists (ACLs).

2.1.3.2 Provisioning

The protocol will not provide any explicit provisioning operations. If it is possible to authenticate or address a principal's calendar resources then they MUST be automatically created if necessary or appropriate

2.1.3.3 Copy/Move

These operations are not yet defined for this version of the CalWS protocol. Both operations raise a number of issues. In particular implementing a move operation through a series of retrievals, insertions and deletions may cause undesirable side-effects. Both these operations will be defined in a later version of this specification.

2.1.3.4 Creating Collections

We will not address the issue of creating collections within the address space. The initial set is created by provisioning.

2.1.3.5 Retrieving collections

This operation is currently undefined. A GET on a collection may fail or return a complete calendar object representing the collection.

2.1.3.6 Setting service and resource properties.

These operations are not defined in this version of the specification. In the future it will be possible to define or set the properties for the service or resources within the service.

2.1.4 CalWS Glossary

2.1.4.1 Hrefs

An href is a URI reference to a resource, for example

```
"http://example.org/user/fred/calendar/event1.ics".
```

The URL above reflects a possible structure for a calendar server. All URLs should be absolute or path-absolute following the rules defined in Error! Reference source not found. Section 8.3.

2.1.4.2 Calendar Object Resource

A calendar object resource is an event, meeting or a task. Attachments are resources but NOT calendar object resources. An event or task with overrides is a single calendar resource entity.

2.1.4.3 Calendar Collection

A folder only allowed to contain calendar object resources.

2.1.4.4 Scheduling Calendar Collection

A folder only allowed to contain calendar resources which is also used for scheduling operations. Scheduling events placed in such a collection will trigger implicit scheduling activity on the server.

2.1.4.5 Principal Home

The collection under which all the resources for a given principal are stored. For example, for principal "fred" the principal home might be "/user/fred/"

2.2 Error conditions

Each operation on the calendar system has a number of pre-conditions and post-conditions that apply.

A "precondition" for a method describes the state of the server that must be true for that method to be performed. A "post-condition" of a method describes the state of the server that must be true after that method has been completed. Any violation of these conditions will result in an error response in the form of a CalWS XML error element containing the violated condition and an optional description. \

Each method specification defines the preconditions that must be satisfied before the method can succeed. A number of post-conditions are generally specified which define the state that must exist after the execution of the operation. Preconditions and post-conditions are defined as error elements in the CalWS XML namespace.

2.2.1 Example: error with CalDAV error condition

```
<?xml version="1.0" encoding="utf-8"
      xmlns:CW="Error! Reference source not found.""
      xmlns:C="http://docs.oasis-open.org/ws-calendar/ns/REST" ?>
<CW:error>
  <C:supported-filter>
    <C:prop-filter name="X-ABC-GUID"/>
  </C:supported-filter>
  <CW:description>Unknown property </CW:description>
</CW:error>
```

3 Properties and link relations

3.1 Property and relation-type URIs

In the XRD entity returned properties and related services and entities are defined by absolute URIs which correspond to the extended relation type defined in **[web linking]** Section 4.2. These URIs do NOT correspond to any real entity on the server and clients should not attempt to retrieve any data at that target.

Certain of these property URIs correspond to CalDAV preconditions. Each URL is prefixed by the CalWS relations and properties namespace http://docs.oasis-open.org/ws-calendar/ns/REST/. Those properties which correspond to CalDAV properties have the additional path element "caldav/", for example

```
http://docs.oasis-open.org/ws-calendar/ns/REST/supported-calendar-data
```

corresponds to

```
CalDAV:supported-calendar-data
```

In addition to those CalDAV properties, the CalWS specification defines a number of other properties and link relations with the URI prefix of http://docs.oasis-open.org/ws-calendar/ns/REST**.**

3.2 supported-features property.

http://docs.oasis-open.org/ws-calendar/ns/REST/supported-features

This property defines the features supported by the target. All resources contained and managed by the service should return this property. The value is a comma separated list containing one or more of the following

- calendar-access - the service supports all MUST requirements in this specification

```
<Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/supported-
features"
          >calendar-access</Property>
```

3.3 max-attendees-per-instance

http://docs.oasis-open.org/ws-calendar/ns/REST/max-attendees-per-instance

Defines the maximum number of attendees allowed per event or task.

3.4 max-date-time

http://docs.oasis-open.org/ws-calendar/ns/REST/max-date-time

Defines the maximum date/time allowed on an event or task

3.5 max-instances

http://docs.oasis-open.org/ws-calendar/ns/REST/max-instances

Defines the maximum number of instances allowed per event or task

3.6 max-resource-size

http://docs.oasis-open.org/ws-calendar/ns/REST/max-resource-size

Provides a numeric value indicating the maximum size of a resource in octets that the server is willing to accept when a calendar object resource is stored in a calendar collection.

3.7 min-date-time

http://docs.oasis-open.org/ws-calendar/ns/REST/min-date-time

Provides a DATE-TIME value indicating the earliest date and time (in UTC) that the server is willing to accept for any DATE or DATE-TIME value in a calendar object resource stored in a calendar collection.

3.8 description

http://docs.oasis-open.org/ws-calendar/ns/REST/description

Provides some descriptive text for the targeted collection.

3.9 timezone-service relation.

http://docs.oasis-open.org/ws-calendar/ns/REST/timezone-service

The location of a timezone service used to retrieve timezone information and specifications. This may be an absolute URL referencing some other service or a relative URL if the current server also provides a timezone service.

```
<Link rel="http://docs.oasis-open.org/ws-calendar/ns/REST/timezone-service"
          href="http://example.com/tz" />
```

3.10 principal-home relation.

http://docs.oasis-open.org/ws-calendar/ns/REST/principal-home

Provides the URL to the user home for the currently authenticated principal.

```
<Link rel=" http://docs.oasis-open.org/ws-calendar/ns/REST/principal-home"
          href="http://example.com/user/fred" />
```

3.11 current-principal-freebusy relation.

http://docs.oasis-open.org/ws-calendar/ns/REST/current-principal-freebusy

Provides the URL to use as a target for freebusy requests for the current authenticated principal.

```
<Link rel=" http://docs.oasis-open.org/ws-calendar/ns/REST/current-principal-freebusy"
          href="http://example.com/freebusy/user/fred" />
```

3.12 principal-freebusy relation.

http://docs.oasis-open.org/ws-calendar/ns/REST/principal-freebusy

Provides the URL to use as a target for freebusy requests for a different principal.

```
<Link rel=" http://docs.oasis-open.org/ws-calendar/ns/REST/principal-freebusy"
          href="http://example.com/freebusy" />
```

3.13 child-collection relation.

http://docs.oasis-open.org/ws-calendar/ns/REST/child-collection

Provides information about a child collections for the target. The href attribute gives the URI of the collection. The element should only have CalWS child elements giving the type of the collection, that is the calWS:collection link property and the CalWS-calendar-collection link property. This allows clients to determine the structure of a hierarchical system by targeting each of the child collections in turn.

The xrd:title child element of the link element provides a description for the child-collection.

```
<Link rel=" http://docs.oasis-open.org/ws-calendar/ns/REST/child-collection"
      href="http://example.com/calWS/user/fred/calendar">
  <Title xml:lang="en">Calendar</Title>
  <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/collection"
            xsi:nil="true" />
```

```
  <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/calendar-
collection"
            xsi:nil="true" />
</Link>
```

3.14 created link property

http://docs.oasis-open.org/ws-calendar/ns/REST/created

Appears within a link relation describing collections or entities. The value is a date-time as defined in **[WS-Calendar]** Section 5.6

```
<Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/created"
          >1985-04-12T23:20:50.52Z</Property>
```

3.15 last-modified property

http://docs.oasis-open.org/ws-calendar/ns/REST/last-modified

Appears within an xrd object describing collections or entities. The value is the same format as would appear in the Last-Modified header and is defined in **[RFC2616],** Section 3.3.1

```
<Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/last-modified"
          >Mon, 12 Jan 1998 09:25:56 GMT</Property>
```

3.16 displayname property

http://docs.oasis-open.org/ws-calendar/ns/REST/displayname

Appears within an xrd object describing collections or entities. The value is a localized name for the entity or collection.

```
<Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/displayname"
          >My Calendar</Property>
```

3.17 timezone property

http://docs.oasis-open.org/ws-calendar/ns/REST/timezone

Appears within an xrd object describing collections. The value is a text timezone identifier.

```
<Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/timezone"
          >America/New_York</Property>
```

3.18 owner property

http://docs.oasis-open.org/ws-calendar/ns/REST/owner

Appears within an xrd object describing collections or entities. The value is a server specific uri.

```
<Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/owner"
         >/principals/users/mike</Property>
```

3.19 collection link property

http://docs.oasis-open.org/ws-calendar/ns/REST/collection

Appears within a link relation describing collections or entities. The property takes no value and indicates that this child element is a collection.

```
<Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/collection"
          xsi:nil="true" />
```

3.20 calendar-collection link property

http://docs.oasis-open.org/ws-calendar/ns/REST/calendar-collection

Appears within a link relation describing collections or entities. The property takes no value and indicates that this child element is a calendar collection.

```
<Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/calendar-collection"
          xsi:nil="true" />
```

3.21 calWS:privilege-set XML element

http://docs.oasis-open.org/ws-calendar/ns/REST/privilege-set

Appears within a link relation describing collections or entities and specifies the set of privileges allowed to the current authenticated principal for that collection or entity.

```
<!ELEMENT calWS:privilege-set (calWS:privilege*)>
<!ELEMENT calWS:privilege ANY>
```

Each privilege element defines a privilege or access right. The following set is currently defined

- calWS: Read - current principal has read access
- calWS: Write - current principal has write access

```
<calWS:privilege-set>
  <calWS:privilege><calWS:read></calWS:privilege>
  <calWS:privilege><calWS:write></calWS:privilege>
</calWS:privilege-set>
```

4 Retrieving Collection and Service Properties

Properties, related services and locations are obtained from the service or from service resources in the form of an XRD document as defined by [XRD-1.0].

Given the URL of a CalWS service a client retrieves the service XRD document through a GET on the service URL with an ACCEPT header specifying application/xrd+xml.

Retrieving resource properties is identical to obtaining service properties, that is, execute a GET on the target URL with an ACCEPT header specifying application/xrd+xml.

The service properties define the global limits and defaults. Any properties defined on collections within the service hierarchy override those service defaults. The service may choose to prevent such overriding of defaults and limits when appropriate.

4.1 Request parameters

- None

4.2 Responses:

- 200: OK
- 403: Forbidden
- 404: Not found

4.3 Example - retrieving server properties:

```
>>Request

GET / HTTP/1.1
Host: example.com
ACCEPT:application/xrd+xml

>>Response
<XRD xmlns="http://docs.oasis-open.org/ns/xri/xrd-1.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <Expires>1970-01-01T00:00:00Z</Expires>
   <Subject>http://example.com/calWS</Subject>
   <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/created"
         >1970-01-01</Property>

   <Link rel=" http://docs.oasis-open.org/ws-calendar/ns/REST/timezone-
service"
         href="http://example.com/tz" />

   <calWS:privilege-set>
   <calWS:privilege><calWS:read></calWS:privilege>
   </calWS:privilege-set>

   <Link rel=" http://docs.oasis-open.org/ws-calendar/ns/REST/principal-home"
         type="collection"
         href="http://example.com/calWS/user/fred">
   <Title xml:lang="en">Fred's calendar home</Title>
   </Link>

   <Link rel=" http://docs.oasis-open.org/ws-calendar/ns/REST/child-
collection"
         type="calendar,scheduling"
```

```
          href="http://example.com/calWS/user/fred/calendar">
   <Title xml:lang="en">Calendar</Title>
   </Link>

   <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/max-
instances"
          >1000</Property>

   <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/max-
attendees-per-instance"
          >100</Property>

</XRD>
```

5 Creating Calendar Object Resources

Creating calendar object resources is carried out by a POST on the parent collection. The body of the request will contain the resource being created. The request parameter "action=create" indicates this POST is a create. The location header of the response gives the URL of the newly created object.

5.1 Request parameters

- action=create

5.2 Responses:

- 201: created
- 403: Forbidden - no access

5.3 Preconditions for Calendar Object Creation

- **calWS:target-exists**: The target of a PUT must exist. Use POST to create entities and PUT to update them.
- **calWS:not-calendar-data:** The resource submitted in the PUT request, or targeted by a COPY or MOVE request, MUST be a supported media type (i.e., iCalendar) for calendar object resources;
- **calWS:invalid-calendar-data:** The resource submitted in the PUT request, or targeted by a COPY or MOVE request, MUST be valid data for the media type being specified (i.e., MUST contain valid iCalendar data);
- **calWS:invalid-calendar-object-resource:** The resource submitted in the PUT request, or targeted by a COPY or MOVE request, MUST obey all restrictions specified in Calendar Object Resources (e.g., calendar object resources MUST NOT contain more than one type of calendar component, calendar object resources MUST NOT specify the iCalendar METHOD property, etc.);
- **calWS:unsupported-calendar-component:** The resource submitted in the PUT request, or targeted by a COPY or MOVE request, MUST contain a type of calendar component that is supported in the targeted calendar collection;
- **calWS:uid-conflict:** The resource submitted in the PUT request, or targeted by a COPY or MOVE request, MUST NOT specify an iCalendar UID property value already in use in the targeted calendar collection or overwrite an existing calendar object resource with one that has a different UID property value. Servers SHOULD report the URL of the resource that is already making use of the same UID property value in the calWS:href element
  <!ELEMENT uid-conflict (calWS:href)>
- **calWS:invalid-calendar-collection-location:** In a COPY or MOVE request, when the Request-URI is a calendar collection, the Destination-URI MUST identify a location where a calendar collection can be created;
- **calWS:exceeds-max-resource-size:** The resource submitted in the PUT request, or targeted by a COPY or MOVE request, MUST have an octet size less than or equal to the value of the CalDAV:max-resource-size property value on the calendar collection where the resource will be stored;
- **calWS:before-min-date-time:** The resource submitted in the PUT request, or targeted by a COPY or MOVE request, MUST have all of its iCalendar DATE or DATE-TIME property values (for each recurring instance) greater than or equal to the value of the CalDAV:min- date-time property value on the calendar collection where the resource will be stored;

- **calWS:after-max-date-time:** The resource submitted in the PUT request, or targeted by a COPY or MOVE request, MUST have all of its iCalendar DATE or DATE-TIME property values (for each recurring instance) less than the value of the CalDAV:max-date-time property value on the calendar collection where the resource will be stored;
- **calWS:too-many-instances:** The resource submitted in the PUT request, or targeted by a COPY or MOVE request, MUST generate a number of recurring instances less than or equal to the value of the CalDAV: max-instances property value on the calendar collection where the resource will be stored;
- **calWS:too-many-attendees-per-instance:** The resource submitted in the PUT request, or targeted by a COPY or MOVE request, MUST have a number of ATTENDEE properties on any one instance less than or equal to the value of the CalDAV:max-attendees-per-instance property value on the calendar collection where the resource will be stored;

5.4 Example - successful POST:

```
>>Request

POST /user/fred/calendar/?action=create HTTP/1.1
Host: example.com
Content-Type: application/xml+calendar; charset="utf-8"
Content-Length: ?

<?xml version="1.0" encoding="utf-8" ?>
<icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
  <vcalendar>
  ...
  </vcalendar>
</icalendar>

>>Response

HTTP/1.1 201 Created
Location: http://example.com/user/fred/calendar/event1.ics
```

5.5 Example - unsuccessful POST:

```
>>Request

POST /user/fred/readcalendar/?action=create HTTP/1.1
Host: example.com
Content-Type: text/text; charset="utf-8"
Content-Length: ?

This is not an xml calendar object

>>Response

HTTP/1.1 403 Forbidden
<?xml version="1.0" encoding="utf-8"
      xmlns:D="DAV:"
      xmlns:C="urn:ietf:params:xml:ns:caldav" ?>
<D:error>
       <C:supported-calendar-data/>
       <D:description>Not an icalendar object</D:description>
</D:error>
```

6 Retrieving resources

A simple GET on the href will return a named resource. If that resource is a recurring event or task with overrides, the entire set will be returned. The desired format is specified in the ACCEPT header. The default form is application/xml+calendar

6.1 Request parameters

- none

6.2 Responses:

- 200: OK
- 403: Forbidden - no access
- 406 The requested format specified in the accept header is not supported.

6.3 Example - successful fetch:

```
>>Request

GET /user/fred/calendar/event1.ics HTTP/1.1
Host: example.com

>>Response

HTTP/1.1 200 OK
Content-Type: application/xml+calendar; charset="utf-8"
Content-Length: ?

<?xml version="1.0" encoding="utf-8" ?>
<icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
  <vcalendar>
    ...
  </vcalendar>
</icalendar>
```

6.4 Example - unsuccessful fetch:

```
>>Request

PUT /user/fred/calendar/noevent1.ics HTTP/1.1
Host: example.com

>>Response

HTTP/1.1 404 Not found
```

7 Updating resources

Resources are updated with the PUT method targeted at the resource href. The body of the request contains a complete new resource which effectively replaces the targeted resource. To allow for optimistic locking of the resource use the if-match header.

When updating a recurring event all overrides and master must be supplied as part of the content.

Preconditions as specified in Section 5.3 are applicable.

7.1 Responses:

- 200: OK
- 304: Not modified - entity was modified by some other request
- 403: Forbidden - no access, does not exist etc. See error response

Example 7-1: Successful Update

```
>>Request

PUT /user/fred/calendar/event1.ics HTTP/1.1
Host: example.com
Content-Type: application/xml+calendar; charset="utf-8"
Content-Length: ?

<?xml version="1.0" encoding="utf-8" ?>
<icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
  <vcalendar>
     ...
  </vcalendar>
</icalendar>

>>Response

HTTP/1.1 200 OK
```

Example 7-2: Unsuccessful Update

```
>>Request

PUT /user/fred/readcalendar/event1.ics HTTP/1.1
Host: example.com
Content-Type: application/xml+calendar; charset="utf-8"
Content-Length: ?

<?xml version="1.0" encoding="utf-8" ?>
<icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
  <vcalendar>
    ...
  </vcalendar>
</icalendar>

>>Response

HTTP/1.1 403 Forbidden
Content-Type: application/xml; charset="utf-8"
Content-Length: xxxx

<?xml version="1.0" encoding="utf-8"
```

```
      xmlns:D="DAV:"
      xmlns:CW=" http://docs.oasis-open.org/ws-calendar/ns/REST/" ?>
<CW:error>
  <CW:target-exists/>
  <CW:description>Target of update must exist</C:description>
</CW:error>
```

8 Deletion of resources

Delete is defined in **[RFC 2616]** Section 9.7. In addition to conditions defined in that specification, servers must remove any references from the deleted resource to other resources. Resources are deleted with the DELETE method targeted at the resource URL. After a successful completion of a deletion a GET on that URL must result in a 404 - Not Found status.

8.1 Delete for Collections

Delete for collections may or may not be supported by the server. Certain collections are considered undeletable. On a successful deletion of a collection all contained resources to any depth must also be deleted.

8.2 Responses:

- 200: OK
- 403: Forbidden - no access
- 404: Not Found

9 Querying calendar resources

Querying provides a mechanism by which information can be obtained from the service through possibly complex queries. A list of icalendar properties can be specified to limit the amount of information returned to the client. A query takes the parts

- Limitations on the data returned
- Selection of the data
- Optional timezone id for floating time calculations.

The current specification uses CalDAV multiget and calendar-query XML bodies as specified in **[RFC 4791]** with certain limitations and differences.

1. The POST method is used for all requests, the action being identified by the outer element.
2. While CalDAV servers generally only support [RFC 5545] and assume that as the default, the delivery format for CalWS will, by default, be [draft-xcal].
3. The CalDAV query allows the specification of a number of DAV properties. Specification of these properties, with the exception of DAV:getetag, is considered an error in CalWS.
4. The CalDAV:propnames element is invalid

With those differences, the CalDAV specification is the normative reference for this operation.

9.1 Limiting data returned

This is achieved by specifying one of the following

- CalDAV:allprop return all properties (some properties are specified as not being part of the allprop set so are not returned)
- CalDAV:prop An element which contains a list of properties to be returned . May only contain DAV:getetag and CalDAV:calendar-data

Of particular interest, and complexity, is the calendar-data property which can contain a time range to limit the range of recurrences returned and/or a list of calendar properties to return.

9.2 Pre/postconditions for calendar queries

The preconditions as defined in in **[RFC 4791]** Section 7.8 apply here. CalDav errors may be reported by the service when preconditions or postconditions are violated.

9.3 Example: time range limited retrieval

This example shows the time-range limited retrieval from a calendar which results in 2 events, one a recurring event and one a simple non-recurring event.

```
>> Request <<

POST /user/fred/calendar/ HTTP/1.1
Host: calWS.example.com
Depth: 1
Content-Type: application/xml; charset="utf-8"
Content-Length: xxxx

<?xml version="1.0" encoding="utf-8" ?>
<C:calendar-query xmlns:D="DAV:"
                  xmlns:C="urn:ietf:params:xml:ns:caldav">
  <D:prop>
    <D:getetag/>
    <C:calendar-data content-type="application/xml+calendar" >
```

```
      <C:comp name="VCALENDAR">
        <C:prop name="VERSION"/>
        <C:comp name="VEVENT">
          <C:prop name="SUMMARY"/>
          <C:prop name="UID"/>
          <C:prop name="DTSTART"/>
          <C:prop name="DTEND"/>
          <C:prop name="DURATION"/>
          <C:prop name="RRULE"/>
          <C:prop name="RDATE"/>
          <C:prop name="EXRULE"/>
          <C:prop name="EXDATE"/>
          <C:prop name="RECURRENCE-ID"/>
        </C:comp>
      </C:comp>
    </C:calendar-data>
  </D:prop>
  <C:filter>
    <C:comp-filter name="VCALENDAR">
      <C:comp-filter name="VEVENT">
        <C:time-range start="20060104T000000Z"
                      end="20060105T000000Z"/>
      </C:comp-filter>
    </C:comp-filter>
  </C:filter>
</C:calendar-query>

>> Response <<

HTTP/1.1 207 Multi-Status
Date: Sat, 11 Nov 2006 09:32:12 GMT
Content-Type: application/xml; charset="utf-8"
Content-Length: xxxx

<?xml version="1.0" encoding="utf-8" ?>
<D:multistatus xmlns:D="DAV:"
               xmlns:C="urn:ietf:params:xml:ns:caldav">
  <D:response>
    <D:href>http://cal.example.com/bernard/work/abcd2.ics</D:href>
    <D:propstat>
      <D:prop>
        <D:getetag>"fffff-abcd2"</D:getetag>
        <C:calendar-data content-type="application/xml+calendar" >
          <xc:icalendar
             xmlns:xc="urn:ietf:params:xml:ns:icalendar-2.0">
   <xc:vcalendar>
     <xc:properties>
      <xc:calscale><text>GREGORIAN</text></xc:calscale>
      <xc:prodid>
       <xc:text>-//Example Inc.//Example Calendar//EN</xc:text>
      </xc:prodid>
      <xc:version><xc:text>2.0</xc:text></xc:version>
     </xc:properties>
     <xc:components>
      <xc:vevent>
       <xc:properties>
        <xc:dtstart>
          <xc:parameters>
            <xc:tzid>US/Eastern<xc:tzid>
          <xc:parameters>
          <xc:date-time>20060102T120000</xc:date-time>
        </xc:dtstart>
        <xc:duration><xc:duration>PT1H</xc:duration></xc:duration>
        <xc:summary>
```

```
          <xc:text>Event #2</xc:text>
        </xc:summary>
        <xc:uid>
         <xc:text>00959BC664CA650E933C892C@example.com</xc:text>
        </xc:uid>
        <xc:rrule>
          <xc:recur>
            <xc:freq>DAILY</xc:freq>
            <xc:count>5</xc:count>
          </xc:recur>
        </xc:rrule>
       </xc:properties>
      </xc:vevent>

      <xc:vevent>
       <xc:properties>
        <xc:dtstart>
          <xc:parameters>
            <xc:tzid>US/Eastern<xc:tzid>
          <xc:parameters>
          <xc:date-time>20060104T140000</xc:date-time>
        </xc:dtstart>
        <xc:duration><xc:duration>PT1H</xc:duration></xc:duration>
        <xc:summary>
         <xc:text>Event #2 bis</xc:text>
        </xc:summary>
        <xc:uid>
         <xc:text>00959BC664CA650E933C892C@example.com</xc:text>
        </xc:uid>
        <xc:recurrence-id>
          <xc:parameters>
            <xc:tzid>US/Eastern<xc:tzid>
          <xc:parameters>
          <xc:date-time>20060104T120000</xc:date-time>
        </xc:recurrence-id>
        <xc:rrule>
          <xc:recur>
            <xc:freq>DAILY</xc:freq>
            <xc:count>5</xc:count>
          </xc:recur>
        </xc:rrule>
       </xc:properties>
      </xc:vevent>

      <xc:vevent>
       <xc:properties>
        <xc:dtstart>
          <xc:parameters>
            <xc:tzid>US/Eastern<xc:tzid>
          <xc:parameters>
          <xc:date-time>20060106T140000</xc:date-time>
        </xc:dtstart>
        <xc:duration><xc:duration>PT1H</xc:duration></xc:duration>
        <xc:summary>
         <xc:text>Event #2 bis bis</xc:text>
        </xc:summary>
        <xc:uid>
         <xc:text>00959BC664CA650E933C892C@example.com</xc:text>
        </xc:uid>
        <xc:recurrence-id>
          <xc:parameters>
            <xc:tzid>US/Eastern<xc:tzid>
          <xc:parameters>
          <xc:date-time>20060106T120000</xc:date-time>
```

```
        </xc:recurrence-id>
        <xc:rrule>
          <xc:recur>
            <xc:freq>DAILY</xc:freq>
            <xc:count>5</xc:count>
          </xc:recur>
        </xc:rrule>
       </xc:properties>
      </xc:vevent>
     </xc:components>
    </xc:vcalendar>
   </xc:icalendar>
          </C:calendar-data>
        </D:prop>
        <D:status>HTTP/1.1 200 OK</D:status>
      </D:propstat>
    </D:response>
    <D:response>
      <D:href>http://cal.example.com/bernard/work/abcd3.ics</D:href>
      <D:propstat>
        <D:prop>
          <D:getetag>"fffff-abcd3"</D:getetag>
          <C:calendar-data content-type="application/xml+calendar" >
            <xcal:icalendar
               xmlns:xc="urn:ietf:params:xml:ns:icalendar-2.0">
    <xc:vcalendar>
     <xc:properties>
      <xc:calscale><text>GREGORIAN</text></xc:calscale>
      <xc:prodid>
       <xc:text>-//Example Inc.//Example Calendar//EN</xc:text>
      </xc:prodid>
      <xc:version><xc:text>2.0</xc:text></xc:version>
     </xc:properties>
     <xc:components>
      <xc:vevent>
       <xc:properties>
        <xc:dtstart>
          <xc:parameters>
            <xc:tzid>US/Eastern<xc:tzid>
          <xc:parameters>
          <xc:date-time>20060104T100000</xc:date-time>
        </xc:dtstart>
        <xc:duration><xc:duration>PT1H</xc:duration></xc:duration>
        <xc:summary>
         <xc:text>Event #3</xc:text>
        </xc:summary>
        <xc:uid>
         <xc:text>DC6C50A017428C5216A2F1CD@example.com</xc:text>
        </xc:uid>
        <xc:rrule>
          <xc:recur>
            <xc:freq>DAILY</xc:freq>
            <xc:count>5</xc:count>
          </xc:recur>
        </xc:rrule>
       </xc:properties>
      </xc:vevent>
     </xc:components>
    </xc:vcalendar>
   </xc:icalendar>
        </C:calendar-data>
      </D:prop>
      <D:status>HTTP/1.1 200 OK</D:status>
    </D:propstat>
```

```
  </D:response>
</D:multistatus>
```

10 Free-busy queries

Free-busy queries are used to obtain free-busy information for a calendar-collection or principals. The result contains information only for events to which the current principal has sufficient access.

When targeted at a calendar collection the result is based only on the calendaring entities contained in that collection. When targeted at a principal free-busy URL the result will be based on all information which affect the principals free-busy status, for example availability.

The possible targets are:

- A calendar collection URL
- The XRD link with relation CalWS/current-principal-freebusy
- The XRD link with relation CalWS/principal-freebusy with a principal given in the request.

The query follows the specification defined in **[FreeBusy Read URL]** with certain limitations. As an authenticated user to the CalWS service scheduling read-freebusy privileges must have been granted. As an unauthenticated user equivalent access must have been granted to unauthenticated access.

Freebusy information is returned by default as xcalendar vfreebusy components, as defined by **[draft-xcal]**. Such a component is not meant to conform to the requirements of VFREEBUSY components in **[RFC 5546]**. The VFREEBUSY component SHOULD conform to section "4.6.4 Free/Busy Component" of **[RFC 5545]**. A client SHOULD ignore the ORGANIZER field..

Since a Freebusy query can only refer to a single user, a client will already know how to match the result component to a user. A server MUST only return a single vfreebusy component.

10.1 ACCEPT header

The Accept header is used to specify the format for the returned data. In the absence of a header the data should be returned as specified in **[draft-xcal]**, that is, as if the following had been specified

```
ACCEPT: application/xml+calendar
```

10.2 URL Query Parameters

None of these parameters are required except for the conditions noted below. Appropriate defaults will be supplied by the server.

10.2.1 start

Default: The default value is left up to the server. It may be the current day, start of the current month, etc.

Description: Specifies the start date for the Freebusy data. The server is free to ignore this value and return data in any time range. The client must check the data for the returned time range.

Format:A profile of an **[RFC3339]** Date/Time. Fractional time is not supported. The server MUST support the expanded version e.g.

```
2007-01-02T13:00:00-08:00
```

It is up to the server to interpret local date/times.

Example:

```
2007-02-03T15:30:00-0800
2007-12-01T10:15:00Z
```

Notes: Specifying only a start date/time without specifying an end-date/time or period should be interpreted as in **[RFC 5545**]. The effective period should cover the remainder of that day.

Date-only values are disallowed as the server cannot determine the correct start of the day. Only UTC or date/time with offset values are permitted.

10.2.2 end

Default: Same as start

Description: Specifies the end date for the Freebusy data. The server is free to ignore this value.

Format: Same as start

Example: Same as start

10.2.3 period

Default: The default value is left up to the server. The recommended value is "P42D".

Description: Specifies the amount of Freebusy data to return. A client cannot specify both a period and an end date. Period is relative to the start parameter.

Format: A duration as defined in section 4.3.6 of **[RFC 5545]**

Example:

```
P42D
```

10.2.4 account

Default: none

Description: Specifies the principal when the request is targeted at the XRD CalWS/principal-freebusy. Specification of this parameter is an error otherwise.

Format: Server specific

Example:

```
fred
/principals/users/jim
user1@example.com
```

10.3 URL parameters - notes

The server is free to ignore the start, end and period parameters. It is recommended that the server return at least 6 weeks of data from the current day.

A client MUST check the time range in the VFREEBUSY response as a server may return a different time range than the requested range.

10.4 HTTP Operations

The server SHOULD return an Etag response header for a successful GET request targeting a Freebusy read URL. Clients MAY use the Etag response header value to do subsequent "conditional" GET requests that will avoid re-sending the Freebusy data again if it has not changed.

10.5 Response Codes

Below are the typical status codes returned by a GET request targeting a Free-busy URL. Note that other HTTP status codes not listed here might also be returned by a server.

- 200 OK
- 302 Found
- 400 Start parameter could not be understood / End parameter could not be understood / Period parameter could not be understood
- 401 Unauthorized
- 403 Forbidden
- 404 The data for the requested principal is not currently available, but may be available later.

- 406 The requested format in the accept header is not supported.
- 410 The data for the requested principal is no longer available
- 500 General server error

10.6 Examples

The following are examples of URLs used to retrieve Free-busy data for a user:

```
http://www.example.com/freebusy/user1@example.com?
start=2007-09-01T00:00:00-08:00&end=2007-09-31T00:00:00-08:00

http://www.example.com/freebusy/user1@example.com?
start=2007-09-01T00:00:00-08:00&end=2007-09-31T00:00:00-08:00

http://www.example.com/freebusy/user1@example.com

http://www.example.com/freebusy?user=user%201@example.com&
start=2008-01-01T00:00:00Z&end=2008-12-31T00:00:00Z
```

Some Request/Response Examples:

A URL with no query parameters:

```
>> Request <<
GET /freebusy/bernard/ HTTP/1.1
Host: www.example.com

>> Response <<
HTTP/1.1 200 OK
Content-Type: application/xml+calendar; charset="utf-8"
Content-Length: xxxx

<xc:icalendar xmlns:xc="urn:ietf:params:xml:ns:icalendar-2.0">
  <xc:vcalendar>
    <xc:properties>
      <xc:calscale><text>GREGORIAN</text></xc:calscale>
      <xc:prodid>
        <xc:text>-//Example Inc.//Example Calendar//EN</xc:text>
      </xc:prodid>
      <xc:version><xc:text>2.0</xc:text></xc:version>
    </xc:properties>
    <xc:components>
      <xc:vfreebusy>
        <xc:properties>
          <xc:uid>
            <xc:text>76ef34-54a3d2@example.com</xc:text>
          </xc:uid>
          <xc:dtstart>
            <xc:date-time>20060101T000000Z</xc:date-time>
          </xc:dtstart>
          <xc:dtend>
            <xc:date-time>20060108T000000Z</xc:date-time>
          </xc:dtend>
          <xc:dtstamp>
            <xc:date-time>20050530T123421Z</xc:date-time>
          </xc:dtstamp>
          <xc:freebusy>
            <xc:parameters>
              <xc:fbtype>BUSYTENTATIVE<xc:fbtype>
            <xc:parameters>
            <xc:period>20060102T100000Z/20060102T120000Z</xc:period>
          </xc:freebusy>
          <xc:freebusy>
            <xc:period>20060103T100000Z/20060103T120000Z</xc:period>
```

```
          </xc:freebusy>
          <xc:freebusy>
            <xc:period>20060104T100000Z/20060104T120000Z</xc:period>
          </xc:freebusy>
          <xc:freebusy>
            <xc:parameters>
              <xc:fbtype>BUSYUNAVAILABLE<xc:fbtype>
            <xc:parameters>
            <xc:period>20060105T100000Z/20060105T120000Z</xc:period>
          </xc:freebusy>
          <xc:freebusy>
            <xc:period>20060106T100000Z/20060106T120000Z</xc:period>
          </xc:freebusy>
      </xc:vfreebusy>
    </xc:components>
  </xc:vcalendar>
<xc:icalendar>
```

A URL with start and end parameters:

```
>> Request <<
GET /freebusy/user1@example.com?start=2007-09-01T00:00:00-08:00&end=2007-09-
31T00:00:00-08:00
HTTP/1.1
Host: www.example.com

>> Response <<
HTTP/1.1 200 OK
Content-Type: application/xml+calendar; charset="utf-8"
Content-Length: xxxx

<xc:icalendar xmlns:xc="urn:ietf:params:xml:ns:icalendar-2.0">
  <xc:vcalendar>
    <xc:properties>
      <xc:calscale><text>GREGORIAN</text></xc:calscale>
      <xc:prodid>
        <xc:text>-//Example Inc.//Example Calendar//EN</xc:text>
      </xc:prodid>
      <xc:version><xc:text>2.0</xc:text></xc:version>
    </xc:properties>
    <xc:components>
      <xc:vfreebusy>
        <xc:properties>
          <xc:uid>
            <xc:text>76ef34-54a3d2@example.com</xc:text>
          </xc:uid>
          <xc:dtstart>
            <xc:date-time>20070901T000000Z</xc:date-time>
          </xc:dtstart>
          <xc:dtend>
            <xc:date-time>20070931T000000Z</xc:date-time>
          </xc:dtend>
          <xc:dtstamp>
            <xc:date-time>20050530T123421Z</xc:date-time>
          </xc:dtstamp>
          <xc:freebusy>
            <xc:period>20070915T230000Z/20070916T010000Z</xc:period>
          </xc:freebusy>
      </xc:vfreebusy>
    </xc:components>
  </xc:vcalendar>
<xc:icalendar>
```

A URL for which the server does not have any data for that user:

```
>> Request <<
```

```
GET /freebusy/user1@example.com?start=2012-12-01T00:00:00-08:00&end=2012-12-
31T00:00:00-08:00
HTTP/1.1
Host: www.example.com

>> Response <<
HTTP/1.1 404 No data
```

11 Conformance

The last numbered section in the specification must be the Conformance section. Conformance Statements/Clauses go here.

Appendix A. Acknowledgments

The following individuals have participated in the creation of this specification and are gratefully acknowledged:

Participants:

Bruce Bartell, Southern California Edison
Brad Benson, Trane
Edward Cazalet, Individual
Toby Considine, University of North Carolina at Chapel Hill
William Cox, Individual
Sharon Dinges, Trane
Mike, Douglass, Rensselaer Polytechnic Institute
Craig Gemmill, Tridium, Inc.
Girish Ghatikar, Lawrence Berkeley National Laboratory
Gerald Gray, Southern California Edison
David Hardin, ENERNOC
Gale Horst, Electric Power Research Institute (EPRI)
Gershon Janssen, Individual
Ed Koch, Akuacom Inc.
Benoit Lepeuple, LonMark International*
Carl Mattocks, CheckMi*
Robert Old, Siemens AG
Alexander Papaspyrou, Technische Universitat Dortmund
Joshua Phillips, ISO/RTO Council (IRC)
Jeremy J. Roberts, LonMark International
David Thewlis, CalConnect

The Calendaring and Scheduling Consortium (CalConnect) TC-XML committee worked closely with WS-Calendar Technical Committee, bridging to developing IETF standards and contributing the services definitions that make up Services in Section 4. The Technical Committee gratefully acknowledges their assistance and cooperation as well. Contributors to TC XML include:

Cyrus Daboo, Apple
Mike Douglass, Rensselaer Polytechnic Institute
Steven Lees, Microsoft
Tong Li, IBM

Appendix B. An Introduction to Internet Calendaring

The WS-Calendar Technical Committee thanks CalConnect for contributing this overview of iCalendar and its use.

B.1 icalendar

B.1.1 History

The iCalendar specification was first produced by the IETF in 1998 as RFC 2445 [1]. Since then it has become the dominant standard for calendar data interchange on the internet and between devices (desktop computers, mobile phones etc.). The specification was revised in 2009 as RFC 5545 [4].

Alongside iCalendar is the iTIP specification (RFC 2446 [2] and revised as RFC 5546[5]) that defines how iCalendar is used to carry out scheduling operations (for example, how an organizer can invite attendees to a meeting and receive their replies). This forms the basis for email-based scheduling using iMIP (the specification that describes how to use iTIP with email - RFC 6047 [3]).

iCalendar itself is a text-based data format. However, an XML format is also available, providing a one-to-one mapping to the text format (draft [7]).

iCalendar data files typically have a .ics file name extension. Most desktop calendar clients can import or export iCalendar data, or directly access such data over the Internet using a variety of protocols.

B.1.2 Data model

The iCalendar data format has a well defined data model. "iCalendar objects" encompass a set of "iCalendar Components" each of which contains a set of "iCalendar properties" and possibly other sub-Components. An iCalendar property consists of a name, a set of optional parameters (specified as "key-value" pairs) and a value.

iCalendar Components include:

"VEVENT" which represents an event

"VTODO" which represents a task or to-do

"VJOURNAL" which represents a journal entry

"VFREEBUSY" which represents periods of free or busy time information

"VTIMEZONE" which represents a timezone definition (timezone offset and daylight saving rules)

"VALARM" is currently the only defined sub-Component and is used to set alarms or reminders on events or tasks.

Properties include:

"DTSTART" which represents a start time for a Component

"DTEND" which represents an end time for a Component

"SUMMARY" which represents a title or summary for a Component

"RRULE" which can specify rules for repeating events or tasks (for example, every day, every week on Tuesdays, etc.)

"ORGANIZER" which represents the calendar user who is organizing an event or assigning a task

"ATTENDEE" which represents calendar users attending an event or assigned a task

In addition to this data model and the pre-defined properties, the specification defines how all those are used together to define the semantics of calendar objects and scheduling. The semantics are basically a set of rules stating how all the Components and properties are used together to ensure that all iCalendar products can work together to achieve good interoperability. For example, a rule requires that all events must have one and only one "DTSTART" property. The most important part of the iCalendar specification

is the semantics of the calendaring model that it represents. The use of text or XML to encode those is secondary.

B.1.3 Scheduling

The iTIP specification defines how iCalendar objects are exchanged in order to accomplish the key task needed to schedule events or tasks. An example of a simple workflow is as follows:

1. To schedule an event, an organizer creates the iCalendar object representing the event and adds calendar users as attendees.
2. The organizer then sends an iTIP "REQUEST" message to all the attendees.
3. Upon receipt of the scheduling message, each attendee can decide whether they want to attend the meeting or not.
4. Each attendee can then respond back to the organizer using an iTIP "REPLY" message indicating their own attendance status.

iTIP supports other types of scheduling messages, for example, to cancel meetings, add new instances to a repeating meeting, etc.

B.1.4 Extensibility

iCalendar was designed to be extensible, allowing for new Components, properties and parameters to be defined as needed. A registry exists to maintain the list of standard extensions with references to their definitions to ensure anyone can use them and work well with others.

B.2 Calendar data access and exchange protocols

B.2.1 Internet Calendar Subscriptions

An Internet calendar subscription is simply an iCalendar data file made available on a web server. Users can use this data in two ways:

- The data can be downloaded from the web server and then imported directly into an iCalendar aware client. This solution works well for calendar data that is not likely to change over time (for example the list of national holidays for the next year).
- Calendar clients that support "direct" subscriptions can use the URL to the calendar data on the web server to download the calendar data themselves. Additionally, the clients can check the web server on a regular basis for updates to the calendar data, and then update their own cached copy of it. This allows calendar data that changes over time to be kept synchronized.

B.2.2 CalDAV

CalDAV is a calendar access protocol and is defined in RFC 4791 [6]. The protocol is based on WebDAV which is an extension to HTTP that provides enhanced capabilities for document management on web servers.

CalDAV is used in a variety of different environments, ranging from very large internet service providers, to large and small corporations or institutions, and to small businesses and individuals.

CalDAV clients include desktop applications, mobile devices and browser-based solutions. It can also be used by "applets", for example, a web page panel that displays a user's upcoming events.

One of the key aspects of CalDAV is its data model. Simply put, it defines a "calendar home" for each calendar user, within which any number of "calendars" can be created. Each "calendar" can contain any number of iCalendar objects representing individual events, tasks or journal entries. This data model ensures that clients and servers can interoperate well.

In addition to providing simple operations to read, write and delete calendar data, CalDAV provides a querying mechanism to allow clients to fetch calendar data matching specific criteria. This is commonly

used by clients to do "time-range" queries, i.e., find the set of events that occur within a given start/end time period.

CalDAV also supports access control allowing for features such as delegated calendars and calendar sharing.

CalDAV also specifies how scheduling operations can be done using the protocol. Whilst it uses the semantics of the iTIP protocol, it simplifies the process by allowing simple calendar data write operations to trigger the sending of scheduling messages, and it has the server automatically process the receipt of scheduling messages. Scheduling can be done with other users on the CalDAV server or with calendar users on other systems (via some form of "gateway").

B.2.3 ActiveSync/SyncML

ActiveSync and SyncML are technologies that allow multiple devices to synchronize data with a server, with calendar data being one of the classes of data supported. These have typically been used for low-end and high-end mobile devices.

B.2.4 CalWS

CalWS refers to a set of web services calendar access APIs developed under a cooperative agreement between The Calendaring and Scheduling Consortium (CalConnect) and OASIS, and being published as a work product of the WS-Calendar Technical Committee. CalWS defines an API to access and manipulate calendar data stored on a server. It follows a similar data model to CalDAV and has been designed to co-exist with a CalDAV service offering the same data.

This specification is part of the CalWS set.

B.2.5 iSchedule

iSchedule is a protocol to allow scheduling between users on different calendaring systems and across different internet domains. It transports iTIP scheduling messages using HTTP between servers. Servers use DNS and various security mechanisms to determine the authenticity of messages received.

It has been specifically designed to be independent of any calendar system in use at the endpoints, so that it is compatible with many different systems. This allows organizations with different calendar systems to exchange scheduling messages with each other, and also allows a single organization with multiple calendar systems (for example due to mergers, or different departmental requirements) to exchange scheduling messages between users of each system.

B.3 References

[1] https://datatracker.ietf.org/doc/rfc2445/ : ‘Internet Calendaring and Scheduling Core Object Specification’

[2] https://datatracker.ietf.org/doc/rfc2446/ :‘iCalendar Transport-Independent Interoperability Protocol’

[3] https://datatracker.ietf.org/doc/rfc6047/ : ‘iCalendar Message-Based Interoperability Protocol’

[4] https://datatracker.ietf.org/doc/rfc5545/ :‘Internet Calendaring and Scheduling Core Object Specification’

[5] https://datatracker.ietf.org/doc/rfc5546/ : ‘iCalendar Transport-Independent Interoperability Protocol’

[6] https://datatracker.ietf.org/doc/rfc4791/ : ‘Calendaring Extensions to WebDAV’

[7] https://datatracker.ietf.org/doc/draft-daboo-et-al-icalendar-in-xml/ : ‘xCal: The XML format for iCalendar’

Appendix C. Revision History

Revision	Date	Editor	Changes Made
ws-calendar-wd19	19-Mar-2011	Toby Considine	Originally contributed by Mike Douglass as part of WS-Calendar v1.0 Specification. See full history in that document.
WD02	13-Feb-2012	Toby Considine	Ported to separate document. “Promoted” all section headers.
WD03	15 Feb-2012	Toby Considine	Added Intro, updated namespaces to meet OASIS standard
WD04	17 February 2012	Toby Considine	Additional namespace clean-up in response to Cover comments. Consisten capitalization of calWS when used as a namespace identifier Clean-up of CalWS discussion in appendix
WD05	17 February 2012	Toby Considine	Types, capitalization, missing XRD reference