CalWS-SOAP SOAP Web Services Protocol for Calendaring

Published Specification

Warning for drafts

This document is not a CalConnect Standard. It is distributed for review and comment, and is subject to change without notice and may not be referred to as a Standard. Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

The Calendaring and Scheduling Consortium, Inc. 2013

:2013

Contents

:2013

Abstract

This document describes standard messages and interactions for service interactions with a system that hosts calendar-based information using SOAP. Hosted information can be either traditional personal and enterprise calendar information or services that support XML payloads developed in conformance with the WS-Calendar specification.

Foreword

This document was subject to public review at the same time as the OASIS WSCalendar publication of CalWS-SOAP and contains the same content.

This document is copyright ©2013 by The Calendaring and Scheduling Consortium and is licensed under the Creative Commons 3.0 Unported License:

http://creativecommons.org/licenses/by/3.0/.

Chair(s) Michael Douglass

Related work This specification is related to:

— RFC 6321 — xCal: iCalendar in XML. http://www.ietf.org/rfc/rfc6321.txt
— WS-Calendar Version 1.0. Latest version.http://docs.oasis-open.org/ws-calendar/ws-calendar/v1.0/ws-calendar-1.0-spec.html

Introduction

The CalWS SOAP protocol is built upon and makes the same assumptions about structure as the CalDAV protocol defined in IETF 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 services will be built without any CalDAV support.

Terminology

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be interpreted as described in IETF RFC 2119.

Namespace

XML namespaces and prefixes used in this standard:

Table 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/soap

1. Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IETF RFC 2119, S. BRADNER. *Key words for use in RFCs to Indicate Requirement Levels*. 1997. RFC Publisher. https://www.rfc-editor.org/info/rfc2119.

IETF RFC 2616, R. FIELDING, J. GETTYS, J. MOGUL, H. FRYSTYK, L. MASINTER, P. LEACH and T. BERNERS-LEE. *Hypertext Transfer Protocol — HTTP/1.1*. 1999. RFC Publisher. https://www.rfc-editor.org/info/rfc2616.

IETF RFC 4791, C. DABOO, B. DESRUISSEAUX and L. DUSSEAULT. *Calendaring Extensions to WebDAV (CalDAV)*. 2007. RFC Publisher. https://www.rfc-editor.org/info/rfc4791.

IETF RFC 6638, C. DABOO and B. DESRUISSEAUX. *Scheduling Extensions to CalDAV*. 2012. RFC Publisher. https://www.rfc-editor.org/info/rfc6638.

IETF RFC 5545, B. DESRUISSEAUX (ed.). *Internet Calendaring and Scheduling Core Object Specification (iCalendar)*. 2009. RFC Publisher. https://www.rfc-editor.org/info/rfc5545.

IETF RFC 5546, C. DABOO (ed.). *iCalendar Transport-Independent Interoperability Protocol (iTIP)*. 2009. RFC Publisher. https://www.rfc-editor.org/info/rfc5546.

IETF RFC 6321, C. DABOO, M. DOUGLASS and S. LEES. *xCal: The XML Format for iCalendar*. 2011. RFC Publisher. https://www.rfc-editor.org/info/rfc6321.

IETF RFC 4790, C. NEWMAN, M. DUERST and A. GULBRANDSEN. *Internet Application Protocol Collation Registry*. 2007. RFC Publisher. https://www.rfc-editor.org/info/rfc4790.

Internet-Draft draft-douglass-timezone-service-04, MIKE DOUGLASS and CYRUS DABOO. *Timezone Service Protocol*. 2012. https://datatracker.ietf.org/doc/html/draft-douglass-timezone-service-04.

CC/S 0903:2009, *Freebusy Read URL V1.0 (CD 0903)*. https://standards.calconnect.org/pubdocs/CD0903%20Freebusy%20Read%20URL%20V1.0.pdf.

SOAP11, Simple Object Access Protocol (SOAP) 1.1, 8 May 2000 http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

WSDL11, Web Services Description Language (WSDL) 1.1, 15 March 2001 http://www.w3.org/TR/2001/NOTE-wsdl-20010315

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-calendarspec-v1.0-cs01.pdf.

2. CalWS Glossary

For the purposes of this document, the following terms and definitions apply.

2.1.
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.

2.2.
UID
The UID of an event is defined in IETF RFC 5545 as a "persistent, globally unique identifier for the calendar component". It is in fact, slightly more complicated in that all overrides to a recurring event have the same UID as the master event. Copies of a meeting invitation sent to attendees

must also have the same UID. In this protocol the UID is the key by which we locate calendar object resources (see above) and any associated overrides within a calendar collection (see below).

2.3.
Collections

A collection is a set of resources which may be entities or other collections. In file systems a collection is commonly referred to as a folder. Collections are referred to by a collection id which is specific to a service and may take any form. For many systems they will be path-like.

2.4.
Calendar Collection

A collection only allowed to contain calendar object resources. The UIDs for components within a calendar collection must be unique. The combination of a calendar collection id and the UID MUST be a unique key within a set of resources made available through this service.

2.5.
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.

2.6.
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/"

2.7.
Change token

This is an opaque token returned to identify the current change status of an entity. Whenever an entity is changed the token will take on a new value. An unchanged token value DOES NOT imply byte-for-byte equality with the stored entity. The service may choose to modify properties under its control, for example last-modification times. However, an entity with an unchanged token can be safely updated by a client holding that token.

CalWS-SOAP SOAP Web Services Protocol for Calendaring

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.

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 ACLs.

3.2. Provisioning

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

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.

3.4. Creating Collections

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

3.5. Retrieving collections

This operation is currently undefined.

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.

4. Basic Calendar Access

This section defines properties, messages and operations sufficient to provide basic access and operations on a calendar store. These are sufficient to store, retrieve and update calendaring entities and to obtain various reports on the current state of the store.

Any service supporting this protocol MUST return a `calendarAccessFeature` element in the `supportedFeatures` property in the `getPropertiesResponse` message as specified in `supportedFeatures`.

4.1. Overview of the CalWS protocol

CalWS operations and data elements are defined in this specification. Many of the operations result in the transmission of data as defined in IETF RFC 5545.

SOAP 1.1 messages consist of three elements: an envelope, header data, and a message body. CalWS request-response elements MUST be enclosed within the SOAP message body. CalWS SOAP messages MUST conform to WT-I-Basic and WS-I-Bind. A single CalWS SOAP message MUST contain only one service request or a single service response).

The basic process for using SOAP for CalWS operations is:

— A system entity acting as a CalWS requester transmits a CalWS request element within the body of a SOAP message to a system entity acting as a CalWS responder. The CalWS requester MUST NOT include more than one CalWS request per SOAP message or include any additional XML elements in the SOAP body (though see Clause 4.10 for multiple messages packaged in one request).
— The CalWS responder MUST return either a CalWS response element within the body of another SOAP message or generate a SOAP fault. The CalWS responder MUST NOT include more than one CalWS response per SOAP message or include any additional XML elements in the SOAP body. If a CalWS responder cannot, for some reason, process a CalWS request, it MUST generate a SOAP fault. (SOAP 1.1 faults and fault codes are discussed in SOAP11, Section 5.1.)

4.1.1. Discovery

CalWS implementers (service providers) MUST provide a WSDL WSDL11 to describe their implementations. This WSDL MAY or may not be made public via a standard discovery mechanism (such as UDDI) or other method.

:2013

In addition, it is REQUIRED that the CalWS implementation include the Properties operation to provide dynamic information regarding CalWS capabilities, options, etc. that are supported.

4.1.2. 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 the execution of a properties operation specifying the target resource. See Clause 4.4

4.1.3. Operations

The following operations are defined by this specification:

— Retrieval and update of service and resource properties
— Creation of a calendar object
— Retrieval of a single calendar object
— Multiget of one or more calendar objects
— Update of a calendar object
— Deletion of a calendar object
— Query
— Free-busy query
— Multiple operations

4.1.4. Calendar Object Resources

The same restrictions apply to Calendar Object Resources as specified in CalDAV IETF RFC 4791, Section 4.2. An additional constraint for CalWS is that no timezone specifications are transferred with the data.

4.1.5. 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. CalWS services may advertise a timezone service (which may be the same service acting as a timezone server) through the server properties object. The timezone service operations are defined in Internet-Draft draft-douglass-timezone-service-04. The service can provide a list of timezone identifiers and aliases.

4.1.6. Error conditions

Each operation on the calendar system has a number of pre-conditions and post-conditions that apply. If any of these are violated the response message will have a status code indicating an error occurred and will contain an error response element providing details.

A "precondition" for a method describes the state of the server that must be true for that method to be performed. A "postcondition" 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 message.

Each method specification defines the preconditions that must be satisfied before the method can succeed. A number of postconditions are generally specified which define the state that must exist after the execution of the operation. Preconditions and postconditions are defined as error elements in the CalWS-SOAP XML namespace, " http://docs.oasis-open.org/ws-calendar/ns/soap".

4.1.6.1. Example: error with error condition

```
<?xml version="1.0" encoding="utf-8"
   xmlns:CW="http://docs.oasis-open.org/ws-calendar/ns/soap" ?>
<CW:error>
  <CW:uidConflict>
    <CW:href>/user/mike/calendar/abcd-0123456789.ics</CW:href>
  </CW:uidConflict>
  <CW:description>Unknown property </CW:description>
</CW:error>
```

4.2. CalWS-SOAP Messages

This section describes the common elements and structure of CalWS-SOAP messages. The conventions followed are shown in Table 2

Table 2 — Field column descriptions

Header	Description	Values	Meaning
Field	Name of the field.		Prefixed with / to indicate a child-relationship Prefixed with # to indicate an attribute
Type	XML schema type		
#	Cardinality of the field	1	One occurrence
		0..1	Zero or one occurrence
		0..*	Zero or more occurrences
		1..*	One or more occurrences
?	Presence	Y	Always required
		N	Optional
		C	Conditional — dependent on the message or other conditions
Description	A short description		

4.2.1. Common Elements and types

The following tables define the base types for requests and responses. All CalWS-SOAP messages and responses are based on these types.

All requests must include an href which specifies the target for the request. There is also an id attribute which will be copied into the response to help identify it.

Table 3 — BaseRequestType elements

Field	Type	#	?	Description
href	string	1	Y	Required in each request to identify the target of the message.
#id	int	1	N	Useful for tying responses to requests.

A response may include an error response element of type ErrorResponseType. This element will be returned in response messages when some form of processing error occurs and provides further information on the error beyond the basic status code.

Table 4 — ErrorResponseType elements

Field	Type	#	?	Description
?	ErrorCodeType	1	Y	One of the error code elements defined below
description	string	0..1	N	Optional descriptive message

4.2.1.1. ErrorCodeType

The following table defines the error codes that may be returned as an element of ErrorCodeType.

Table 5 — ErrorCodeType definitions

Field	Type	Description
forbidden	ForbiddenType	Attempted to carry out a forbidden operation.
targetExists	TargetExistsType	
targetDoesNotExist	TargetDoesNotExistType	The supplied href does not reference an existing resource.
targetNotEntity	TargetNotEntityType	The supplied href does not target an entity. For example a fetch item was attempted against a collection.
notCalendarData	NotCalendarDataType	The supplied entity is not calendar data.
invalidCalendarData	InvalidCalendarDataType	The supplied entity does not represent valid calendar data.
invalidCalendarObjectResource	InvalidCalendarObjectResourceType	The supplied entity does not represent valid calendar data.
unsupportedCalendarComponent	UnsupportedCalendarComponentType	Indicates that the calendar collection does not

Field	Type	Description
		accept components of the type the client is attempting to store. The accepted component types can be determined by examining the calendar collection properties.
invalidCalendarCollectionLocation	InvalidCalendarCollectionLocationType	Error indicating at least one of two conditions: 1) The server does not allow the creation of calendar collections at the given location in its namespace, or 2) The parent collection of the Request-URI exists but cannot accept members
exceedsMaxResourceSize	ExceedsMaxResourceSizeType	Error indicating that the total size of the event or task is too

Field	Type	Description
		large. The maximum size is set by the target system and can be determined from the properties.
beforeMinDateTime	BeforeMinDateTimeType	Error indicating that the start or end of an event or task is too far into the past.
		The minimum date is set by the target system and can be determined from the properties.
afterMaxDateTime	AfterMaxDateTimeType	Error indicating that the start or end of an event or task is too far into the future.
		The maximum date is set by the target system and can be determined from the properties.
tooManyInstances	TooManyInstancesType	Error indicating that a recurring event has

Field	Type	Description
		too many instances. The maximum number is set by the target system and can be determined from the properties.
tooManyAttendeesPerInstance	TooManyAttendeesPerInstanceType	Error indicating that a scheduling message has too many attendees. The maximum number is set by the target system and can be determined from the properties.
partialSuccess	PartialSuccessType	Indicates that a `MultiOpType` operation was partially successful. Returned when the operation is marked as non-atomic and one or more sub-operations failed. The entire response needs to be examined to determine

Field	Type	Description
		failing operations.
missingChangeToken	MissingChangeTokenType	An operation was attempted which required a change token but none was supplied.

Note that it appears that the marshalling or demarshalling should handle this as the token is required. It doesn't. |
| mismatchedChangeToken | MismatchedChangeTokenType | An update operation was attempted with a change token value which does not match that held by the service. The client must refetch the entity to refresh its cached value and token.

Note that matching of tokens is a server responsibility. The token is opaque to the client but probably structured to the |

Field	Type	Description
		server. Certain non-conflicting updates may be allowed even if the token has changed.
invalidFilter	InvalidFilterType	
uidConflict	UidConflictType	An attempt was made to store an entity which would result in more than one entity having equal uids. The entity uid must be unique within a collection. Recurring event or task overrides have the same uid and are considered part of a single entity.

4.2.1.2. BaseResponseType

Table 6 — BaseResponseType elements

Field	Type	#	?	Description
#id	int	1	N	Copied over from the request
status	StatusType	1	Y	Give the overall status of the response
message	string	0..1	N	Optional explanatory message
errorResponse	ErrorCodeType	0..1	N	Required for a status of Error.

4.3. Properties

The getPropertiesResponse message contains 0 or more properties defined below. Some properties apply to the service as a whole while others apply only to the targeted resource. The targeted resource may have property values which override those for the service. For example, the

:2013

timezone identifier for a particular collection may differ from the default timezone identifier for the system.

Each property is an XML complex type based on the `GetPropertiesBasePropertyType`.

4.3.1. `childCollection`

Provides information about a child collections for the target.

Table 7 — `ChildCollectionType` fields

Field	Type	#	?	Description
href	string	1	Y	The URI of the collection.
collection	CollectionType	1	Y	This is a collection
calendarCollection	CalendarCollectionType	0..1	C	If present this is a calendar collection

See [Clause 4.3.14](#) for descriptions of `CollectionType` and `CalendarCollectionType`.

4.3.2. `creationDateTime`

This property MAY be returned for the service and SHOULD be returned for any targeted resource.

Table 8 — `CreationDateTimeType` fields

Field	Type	#	?	Description
dateTime	dateTime	1	Y	Creation date/time of the resource

4.3.3. `displayName`

This property SHOULD be returned for any targeted resource.

Table 9 — `DisplayNameType` fields

Field	Type	#	?	Description
string	string	1	Y	The displayable name.

4.3.4. `lastModifiedDateTime`

This property MAY be returned for the service and SHOULD be returned for any targeted resource.

Table 10 — `LastModifiedDateTimeType` fields

Field	Type	#	?	Description
dateTime	dateTime	1	Y	Last modified date/time of the resource

4.3.5. `maxAttendeesPerInstance`

This property SHOULD be returned for the service and MAY be returned for any targeted collection resource.

Table 11 — `MaxAttendeesPerInstanceType` fields

Field	Type	#	?	Description
integer	integer	1	Y	The maximum number of attendees allowed per event or task instance.

4.3.6. `maxDateTime`

This property SHOULD be returned for the service and MAY be returned for any targeted collection resource.

Table 12 — MaxDateTimeType fields

Field	Type	#	?	Description
dateTime	dateTime	1	Y	The maximum date and time for an event.

4.3.7. maxInstances

This property SHOULD be returned for the service and MAY be returned for any targeted collection resource.

Table 13 — MaxInstancesType fields

Field	Type	#	?	Description
integer	integer	1	Y	The maximum number of instances for a recurring event.

4.3.8. maxResourceSize

This property SHOULD be returned for the service and MAY be returned for any targeted collection resource.

Table 14 — MaxResourceSizeType fields

Field	Type	#	?	Description
integer	integer	1	Y	An integer value defining 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.

4.3.9. minDateTime

This property SHOULD be returned for the service and MAY be returned for any targeted collection resource.

Table 15 — MinDateTimeType fields

Field	Type	#	?	Description
dateTime	dateTime	1	Y	The minimum date and time for an event.

4.3.10. principalHome

This property SHOULD be returned for the service and MAY be returned for any targeted collection resource.

Table 16 — PrincipalHomeType fields

Field	Type	#	?	Description
string	string	1	Y	The home path of the currently authenticated user.

4.3.11. resourceDescription

Provides some descriptive text for the targeted collection.

Table 17 — ResourceDescriptionType fields

Field	Type	#	?	Description
string	string	1	Y	The descriptive text.

4.3.12. resourceOwner

This property SHOULD be returned for any targeted resource.

:2013

Table 18 — ResourceownerType fields

Field	Type	#	?	Description
string	string	1	Y	The principal URL of the resource owner.

4.3.13. resourceTimezoneId

This property SHOULD be returned for the service and MAY be returned for any targeted collection resource.

Table 19 — ResourceTimezoneIdType fields

Field	Type	#	?	Description
string	string	1	Y	The timezone identifier.

4.3.14. resourceType

Provides information about a targeted resource.

Table 20 — ResourceTypeType fields

Field	Type	#	?	Description
href	string	1	Y	The URI of the collection.
collection	CollectionType	0..1	C	If present this is a collection
calendarCollection	CalendarCollectionType	0..1	C	If present this is a calendar collection
inbox	InboxType	0..1	C	If present this is a scheduling inbox
outbox	OutboxType	0..1	C	If present this is a scheduling outbox
inbox	InboxType	0..1	C	If present this is a scheduling inbox
xresource	XresourceType	0..1	C	If present provides further type information.

All the child types are empty elements with the exception of XresourceType.

Table 21 — XresourceType fields

Field	Type	#	?	Description
string	string	1	Y	Extra information.

4.3.15. supportedCalendarComponentSet

This property identifies which component types the service is prepared to store. The allowable components may be different for different targets on the same service.

Table 22 — SupportedCalendarComponentSetType fields

Field	Type	#	?	Description
Any valid iCalendar component name	xcal: BaseComponentType	0..n	C	One or more empty iCalendar components.

4.3.16. supportedFeatures

This property SHOULD be returned for the service and MAY be returned for any targeted collection resource.

The property shows what protocol features are supported by the server.

Table 23 — SupportedFeaturesType fields

Field	Type	#	?	Description
calendarAccessFeature	CalendarAccessFeatureType	1	Y	Indicates the service supports this protocol.

4.3.17. `timezoneServer`

This property SHOULD be returned for the service and MAY be returned for any targeted collection resource.

<div align="center">

Table 24 — `TimezoneServerType` fields

</div>

Field	Type	#	?	Description
string	string	1	Y	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.

4.3.18. CalWS:privilege-set XML element

http://docs.oasis-open.org/ns/wscal/calws: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>
```

4.4. Retrieving Collection and Service Properties

The CalWS-SOAP getProperties request is used to fetch properties. The `href` can target the service with a path of "/" or any entity within the service.

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. The tables below show the fields for request and response.

<div align="center">

Table 25 — `GetPropertiesType` fields

</div>

Field	Type	#	?	Description
href	string	1	Y	Identify the target of the request. "/" for the service.

<div align="center">

Table 26 — `GetPropertiesResponseType` fields

</div>

Field	Type	#	?	Description
href	string	1	Y	Identify the target of the request. "/" for the service.
?	GetPropertiesBasePropertyType	0..n	C	0 or more properties of the targeted resource

4.4.1. Example — retrieving server properties

```
>>Request

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
```

```
:2013

  <SOAP-ENV:Body>
    <ns2:getProperties xmlns:ns2="http://docs.oasis-open.org/ws-calendar/ns/soap"
        xmlns:ns3="urn:ietf:params:xml:ns:icalendar-2.0">
      <ns2:href>/</ns2:href>
    </ns2:getProperties>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

>>Response

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header />
  <SOAP-ENV:Body>
    <ns2:getPropertiesResponse
      xmlns:ns2="http://docs.oasis-open.org/ws-calendar/ns/soap"
      xmlns:ns4="urn:ietf:params:xml:ns:icalendar-2.0"
      id="0" >
      <ns2:href>/</ns2:href>
      <ns2:lastModifiedDateTime>
        <ns2:dateTime>2012-01-04T18:21:14Z</ns2:dateTime>
      </ns2:lastModifiedDateTime>
      <ns2:supportedCalendarComponentSet>
        <ns4:vevent />
        <ns4:vtodo />
        <ns4:vavailability />
      </ns2:supportedCalendarComponentSet>
      <ns2:resourceType>
        <ns2:collection />
      </ns2:resourceType>
      <ns2:supportedFeatures>
        <ns2:calendarAccessFeature />
      </ns2:supportedFeatures>
      <ns2:maxInstances>
        <ns2:integer>1000</ns2:integer>
      </ns2:maxInstances>
      <ns2:maxResourceSize>
        <ns2:integer>100000</ns2:integer>
      </ns2:maxResourceSize>
    </ns2:getPropertiesResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

4.5. Creating Calendar Object Resources

Creating calendar object resources is carried out by using a CalWS-SOAP addItem request targeted at the parent collection and containing the resource to be created. The response will contain the `href` of the newly created object.

The iCalendar entity in the request MUST contain only a single calendaring entity with any related overrides.

Table 27 — `AddItemType` fields

Field	Type	#	?	Description
href	string	1	Y	Identify the target of the request.
icalendar	xcal:IcalendarType	1	Y	The entity to be created

The service will respond with an `AddItemResponseType` giving either the `href` and change token of the new entity or an error response.

Table 28 — `AddItemResponseType` additional fields

Field	Type	#	?	Description
href	string	0..1	N	Href of the new entity for a successful request.
changeToken	string	0..1	N	Change token for the new entity

4.5.1. Preconditions for Calendar Object Creation

— **CalWS:target-exists**: The entity already exists.
— **CalWS:not-calendar-data**: The resource submitted MUST be a supported media type (i.e., iCalendar) for calendar object resources;
— **CalWS:invalid-calendar-data**: The resource submitted 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 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 request MUST contain a type of calendar component that is supported in the targeted calendar collection;
— **CalWS:uid-conflict**: The resource submitted in the 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:exceeds-max-resource-size**: The resource submitted in the 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 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 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 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 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.

4.5.2. Example — successful `addItem`

```
>>Request

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:addItem xmlns:ns2="http://docs.oasis-open.org/ws-calendar/ns/soap"
                 xmlns:ns3="urn:ietf:params:xml:ns:icalendar-2.0">
      <ns2:href>/user/douglm/calendar</ns2:href>
      <ns3:icalendar>
        <ns3:vcalendar>
          <ns3:components>
            <ns3:vevent>
```

```
                  <ns3:properties>
                    <ns3:uid>
                       <ns3:text>1302064354993</ns3:text>
                    </ns3:uid>
                    <ns3:summary>
                       <ns3:text>try this</ns3:text>
                    </ns3:summary>
                    <ns3:dtstart>
                       <ns3:date-time>20110406T150000Z</ns3:date-time>
                    </ns3:dtstart>
                    <ns3:dtend>
                       <ns3:date-time>20110406T160000Z</ns3:date-time>
                    </ns3:dtend>
                  </ns3:properties>
               </ns3:vevent>
             </ns3:components>
           </ns3:vcalendar>
         </ns3:icalendar>
      </ns2:addItem>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

>>Response

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
   <SOAP-ENV:Header/>
   <SOAP-ENV:Body>
     <ns2:addItemResponse xmlns:ns2="http://docs.oasis-open.org/ws-calendar/ns/
soap"
                          xmlns:ns3="urn:ietf:params:xml:ns:icalendar-2.0">
      <ns2:status>OK</ns2:status>
      <ns2:href>/user/douglm/calendar/1302064354993.ics</ns2:href>
      <ns2:changeToken>"20110406T155741Z-0"</ns2:changeToken>
     </ns2:addItemResponse>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

4.6. Retrieving resources

Fetching calendar object resources is carried out by using a CalWS-SOAP fetchItem request with an href specifying the entity to be fetched. The response will contain the calendaring entity with any related overrides.

Table 29 — `FetchItemType` fields

Field	Type	#	?	Description
href	string	1	Y	Identify the target of the request.

The service will respond with a FetchItemResponseType containing either the change token, its href and the entity or an error response.

Table 30 — `FetchItemResponseType` additional fields

Field	Type	#	?	Description
changeToken	string	0..1	N	The change token for the fetched entity
href	string	1	Y	Identify the entity.
icalendar	xcal:IcalendarType	0..1	N	The fetched entity

4.6.1. Example — successful `fetchItem`

>>Request

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:fetchItem xmlns:ns2="http://docs.oasis-open.org/ws-calendar/ns/soap"
                   xmlns:ns3="urn:ietf:params:xml:ns:icalendar-2.0">
      <ns2:href>/user/douglm/calendar/1302105461170.ics</ns2:href>
    </ns2:fetchItem>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

>>Response

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:fetchItemResponse xmlns:ns2="http://docs.oasis-open.org/ws-calendar/ns/soap"
                           xmlns:ns3="urn:ietf:params:xml:ns:icalendar-2.0">
      <ns2:status>OK</ns2:status>
      <ns2:changeToken>"20110406T155741Z-0"</ns2:changeToken>
      <ns2:href>/user/douglm/calendar/1302105461170.ics</ns2:href>
      <ns3:icalendar>
        <ns3:vcalendar>
          <ns3:properties>
            <ns3:prodid>
              <ns3:text>//Bedework.org//BedeWork V3.7//EN</ns3:text>
            </ns3:prodid>
            <ns3:version>
              <ns3:text>2.0</ns3:text>
            </ns3:version>
          </ns3:properties>
          <ns3:components>
            <ns3:vevent>
              <ns3:properties>
                <ns3:created>
                  <ns3:utc-date-time>20110406T155741Z</ns3:utc-date-time>
                </ns3:created>
                <ns3:dtend>
                  <ns3:date-time>20110406T160000Z</ns3:date-time>
                </ns3:dtend>
                <ns3:dtstamp>
                  <ns3:utc-date-time>20110406T155741Z</ns3:utc-date-time>
                </ns3:dtstamp>
                <ns3:dtstart>
                  <ns3:date-time>20110406T150000Z</ns3:date-time>
                </ns3:dtstart>
                <ns3:last-modified>
                  <ns3:utc-date-time>20110406T155741Z</ns3:utc-date-time>
                </ns3:last-modified>
                <ns3:summary>
                  <ns3:text>try this</ns3:text>
                </ns3:summary>
                <ns3:uid>
```

:2013

```
                    <ns3:text>1302105461170</ns3:text>
                </ns3:uid>
            </ns3:properties>
          </ns3:vevent>
        </ns3:components>
      </ns3:vcalendar>
    </ns3:icalendar>
  </ns2:fetchItemResponse>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

4.6.2. Example — unsuccessful `fetchItem`

>>Request

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:fetchItem xmlns:ns2="http://docs.oasis-open.org/ws-calendar/ns/soap"
                   xmlns:ns3="urn:ietf:params:xml:ns:icalendar-2.0">
      <ns2:href>/user/douglm/calendar/nosuchevent.ics</ns2:href>
    </ns2:fetchItem>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

>>Response

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:fetchItemResponse xmlns:ns2="http://docs.oasis-open.org/ws-calendar/ns/
soap"
                           xmlns:ns3="urn:ietf:params:xml:ns:icalendar-2.0">
      <ns2:status>Error</ns2:status>
      <ns2:errorResponse>
        <ns2:targetDoesNotExist/>
      </ns2:errorResponse>
    </ns2:fetchItemResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

4.7. Updating resources

Calendar entity updates apply changes to a data model which has the form:

— An iCalendar element contains…
— a single vCalendar element which contains…
— one or more calendaring components, event, task etc. each of which contain…
— zero or more components, alarms etc. or one or more properties each of which contains…
— zero or more parameters and one or more values.

Thus we have a nested structure which does recurse to a limited extent and looks like

```
<icalendar>
  <vcalendar>
    <components>
      <vevent>
```

```
            <properties>
              <uid>
                <text>1302064354993-a</text>
              </uid>
              <summary>
                <text>try this</text>
              </summary>
              <dtstart>
                <date-time>2011-07-18T15:00:00Z</date-time>
              </dtstart>
              <dtend>
                <date-time>2011-07-18T16:00:00Z</date-time>
              </dtend>
            </properties>
          </vevent>
        </components>
      </vcalendar>
    </icalendar>
```

The update approach described here only allows for updating a single calendar entity, though that entity may consist of more than one component, for example an override to a repeating event.

Resources are updated with the CalWS-SOAP updateItem request. The request contains the href of the entity to be updated, the current change token for that entity and the updates. The updates take the form of nested selections of an element from the current level in the data. The outermost selection is always for a vCalendar element — we ignore the iCalendar element. Nested within that outer selection is one for the components element followed by selections on the entity, event, task etc and so on.

Only 3 kinds of update may be applied at any point:

— Remove — components, properties or parameters
— Add — components, properties or parameters
— Change — property or parameter values

Removals MUST be processed ahead of additions.

Preconditions as specified in Preconditions for Calendar Object Creation are applicable. The response will indicate success or failure of the update. If the change token value does not match that held by the service a mismatchedChangeToken error status will be returned. The client should re-fetch the entity to refresh its cache and then retry the update based on the new entity values and change token.

Table 31 — UpdateItemType fields

Field	Type	#	?	Description
href	string	1	Y	Identify the target of the request.
changeToken	string	1	Y	The change token held by the client for that entity
select	ComponentSelectionType	1..*	Y	Must select vCalendar

The ComponentsSelectionType contains three repeating child elements. The first allows for selection of nested components which can then be updated. The next allows addition of entire components and the last allows for the removal of components.

Table 32 — ComponentsSelectionType fields

Field	Type	#	?	Description
component	ComponentSelectionType	0..1	N	Used to match against a component in the target
remove	ComponentReferenceType	0..1	N	Supplies components to remove
add	ComponentReferenceType	0..1	N	Species components to add

The `PropertiesSelectionType` follows the same pattern, selecting properties to update, add or remove.

Table 33 — `PropertiesSelectionType` fields

Field	Type	#	?	Description
property	PropertySelectionType	0..1	N	Used to match against a property in the target
remove	PropertyReferenceType	0..1	N	Supplies properties to remove
add	PropertyReferenceType	0..1	N	Species properties to add

To complete that pattern there is also a `ParametersSelectionType` used to select property parameters for update or removal and to supply new parameters.

Table 34 — `ParametersSelectionType` fields

Field	Type	#	?	Description
parameter	ParameterSelectionType	0..1	N	Used to match against a parameter in the target
remove	ParameterReferenceType	0..1	N	Supplies parameters to remove
add	ParameterReferenceType	0..1	N	Species parameters to add

Each of these refers to a reference type. These either provide a complete entity for addition or identify the entity for removal. The three reference types are:

Table 35 — `ComponentReferenceType` fields

Field	Type	#	?	Description
Any valid iCalendar component name	xcal: BaseComponentType	1	Y	Either a complete component or sufficient to identify it.

Table 36 — `PropertyReferenceType` fields

Field	Type	#	?	Description
Any valid iCalendar property name	xcal: BasePropertyType	1	Y	Either a complete property or sufficient to identify it or provide a new value, depending on usage.

Table 37 — `ParameterReferenceType` fields

Field	Type	#	?	Description
Any valid iCalendar parameter name	xcal: BaseParameterType	1	Y	Either a complete parameter or sufficient to identify it or provide a new value, depending on usage.

To complete the picture we have three selection types for component, property and parameter. Each of these identifies the entity to be updated, possible selections of the sub-elements and a possible change to values.

`ComponentSelectionType` contains three child elements. The first is any valid iCalendar component element which is to be matched at the current level.

The optional properties selection allows selection and possible updates to the properties of the component. An iCalendar properties element cannot take a value so the only updates possible are addition and removal of properties. Nested properties may be selected for updates.

The optional components selection allows selection and possible updates to the nested iCalendar components element of the component. An iCalendar components element cannot take a value so the only updates possible are addition and removal of components. Nested components may be selected for updates.

Table 38 — `ComponentSelectionType` fields

Field	Type	#	?	Description
Any valid iCalendar component name	xcal:VcalendarType xcal:BaseComponentType	1	Y	Used to match against an element in the target
properties	PropertiesSelectionType	0..1	N	To match the properties element
components	ComponentsSelectionType	0..1	N	To match the components element

`PropertySelectionType` contains three child elements. The first is any valid iCalendar property element which is to be matched at the current level.

The optional parameters selection allows selection and possible updates to the parameters of the property. The optional change element allows a change to the value of the property. The new value is specified by supplying an iCalendar property with the desired value(s). Any parameters will be ignored.

Table 39 — `PropertySelectionType` fields

Field	Type	#	?	Description
Any valid iCalendar property name	`xcal:BasePropertyType`	1	Y	Used to match against an element in the target
parameters	`ParametersSelectionType`	0..1	N	To match the parameters element
change	`PropertyReferenceType`	0..1	N	To provide a new value

Lastly, there is the `ParameterSelectionType` which contains two child elements. The first is any valid iCalendar parameter element which is to be matched at the current level. The optional change element allows a change to the value of the parameter. The new value is specified by supplying an iCalendar parameter with the desired value(s).

Table 40 — `ParameterSelectionType` fields

Field	Type	#	?	Description
Any valid iCalendar parameter name	`xcal:BaseParameterType`	1	Y	Used to match against an element in the target
change	`ParameterReferenceType`	0..1	N	To provide a new value

For a successful update the service will respond with a `UpdateItemResponseType` containing the status and the new change token.

Table 41 — `UpdateItemResponseType` additional fields

Field	Type	#	?	Description
changeToken	string	0..1	N	The new change token for the updated entity

The change token value should be used to replace the value held by the client.

4.7.1. Change tokens and concurrent updates

The change token is used to allow a service to determine whether or not it is safe to carry out an update requested by the client. The change token should be opaque to the client but will probably in fact be a structured value. Calendaring transactions have some special characteristics which make it desirable to allow certain non-conflicting updates to take place while other changes are taking place. For example, meeting requests with a large number of attendees can be frequently updated by the server as a result of attendee participation status changes. If we use an unstructured change token to represent all changes this can make it very difficult to update an event while those participation status changes are being made. If, on the other hand, the token has a section indicating that only participation status changes have been made, then other changes can take place. For a reference on implementing such a token see "Avoiding Conflicts when Updating Scheduling Object Resources" in [IETF RFC 6638](). This describes the use of a schedule-tag.

4.7.2. Example — successful update

The event to be updated is represented by the following XML.

```
<ns3:icalendar>
  <ns3:vcalendar>
    <ns3:components>
      <ns3:vevent>
        <ns3:properties>
          <ns3:uid>
            <ns3:text>1302064354993-a</ns3:text>
```

:2013

```
        </ns3:uid>
        <ns3:summary>
          <ns3:text>try this</ns3:text>
        </ns3:summary>
        <ns3:dtstart>
          <ns3:date-time>2011-07-18T15:00:00Z</ns3:date-time>
        </ns3:dtstart>
        <ns3:dtend>
          <ns3:date-time>2011-07-18T16:00:00Z</ns3:date-time>
        </ns3:dtend>
      </ns3:properties>
    </ns3:vevent>
  </ns3:components>
 </ns3:vcalendar>
</ns3:icalendar>
```

In the following example we make the following changes to the above event:

— Change the summary
— Change the dtstart — add a tzid and change the value to local time
— Add some categories

We first select an event by specifying the uid value and then, from that event, we select the properties, then select and change the appropriate properties.

>>Request

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:updateItem xmlns:ns2="http://docs.oasis-open.org/ws-calendar/ns/soap"
                    xmlns:ns3="urn:ietf:params:xml:ns:icalendar-2.0">
      <ns2:href>/user/douglm/calendar/1302064354993-a.ics</ns2:href>
      <ns2:changeToken>"20110802T032608Z-0" null</ns2:changeToken>
      <ns2:select>
        <ns3:vcalendar/>
          <ns2:components>
            <ns2:component>
              <ns3:vevent>
                <ns3:properties>
                  <ns3:uid>
                    <ns3:text>1302064354993-a</ns3:text>
                  </ns3:uid>
                </ns3:properties>
              </ns3:vevent>
            <ns2:properties>
              <ns2:property>
                <ns3:dtstart>
                  <ns3:date-time>2011-07-18T15:00:00Z</ns3:date-time>
                </ns3:dtstart>
                <ns2:parameters>
                  <ns2:add>
                    <ns3:tzid>
                      <ns3:text>America/New_York</ns3:text>
                    </ns3:tzid>
                  </ns2:add>
                </ns2:parameters>
                <ns2:change>
                  <ns3:dtstart>
```

```
                    <ns3:date-time>2011-07-18T11:00:00</ns3:date-time>
                  </ns3:dtstart>
                </ns2:change>
              </ns2:property>
              <ns2:property>
                <ns3:summary>
                  <ns3:text>try this</ns3:text>
                </ns3:summary>
                <ns2:change>
                  <ns3:summary>
                    <ns3:text>A changed summary - again and again and again</ns3:
text>
                  </ns3:summary>
                </ns2:change>
              </ns2:property>
              <ns2:add>
                <ns3:categories>
                  <ns3:text>newcategory-2</ns3:text>
                  <ns3:text>resources</ns3:text>
                  <ns3:text>paper</ns3:text>
                </ns3:categories>
              </ns2:add>
            </ns2:properties>
          </ns2:component>
        </ns2:components>
      </ns2:select>
    </ns2:updateItem>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

>>Response

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:updateItemResponse xmlns:ns2="http://docs.oasis-open.org/ws-calendar/ns/
soap"
                            xmlns:ns3="urn:ietf:params:xml:ns:icalendar-2.0"id=
"0">
      <ns2:status>OK</ns2:status>
    </ns2:updateItemResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

4.7.3. Other updates

Based on the example above we present some XML fragments for different kinds of update. These include:

— Addition of properties
— Removal of properties
— Addition of parameters to properties
— Removal of parameters from properties
— Changing parameter values.

The examples all start with the selection of the vevent properties element. First we have the XML for the addition of a `tzid` to the start date/time. Here we select the `dtstart`, then the parameters element then add a `tzid` parameter and change the value of the date and time

:2013

```
<ns2:properties>
  <ns2:property>
    <ns3:dtstart>
      <ns3:date-time>2011-07-18T15:00:00Z</ns3:date-time>
    </ns3:dtstart>
    <ns2:parameters>
      <ns2:add>
        <ns3:tzid>
          <ns3:text>America/New_York</ns3:text>
        </ns3:tzid>
      </ns2:add>
    </ns2:parameters>
    <ns2:change>
      <ns3:dtstart>
        <ns3:date-time>2011-07-18T11:00:00</ns3:date-time>
      </ns3:dtstart>
    </ns2:change>
  </ns2:property>
</ns2:properties>
```

In this example we add two categories to the event.

```
<ns2:properties>
  <ns2:add>
    <ns3:categories>
      <ns3:text>paper</ns3:text>
    </ns3:categories>
  </ns2:add>
  <ns2:add>
    <ns3:categories>
      <ns3:text>resources</ns3:text>
    </ns3:categories>
  </ns2:add>
</ns2:properties>
```

In this example we add a `duration` and remove the `dtend`.

```
<ns2:properties>
  <ns2:remove>
    <ns3:dtend>
      <ns3:date-time>2011-07-18T16:00:00Z</ns3:date-time>
    </ns3:dtend>
  </ns2:remove>
  <ns2:add>
    <ns3:duration>
      <ns3:duration>PT1H</ns3:duration>
    </ns3:duration>
  </ns2:add>
</ns2:properties>
```

In this example we change the `dtstart` timezone identifier.

```
<ns2:properties>
  <ns2:property>
    <ns3:dtstart>
      <ns3:parameters>
        <ns3:tzid>
          <ns3:text>America/New_York</ns3:text>
        </ns3:tzid>
      </ns3:parameters>
      <ns3:date-time>2011-07-18T11:00:00</ns3:date-time>
    </ns3:dtstart>
```

```
  <ns2:parameters>
    <ns2:parameter>
      <ns3:tzid>
        <ns3:text>America/New_York</ns3:text>
      </ns3:tzid>
      <ns2:change>
        <ns3:tzid>
          <ns3:text>America/Montreal</ns3:text>
        </ns3:tzid>
      </ns2:change>
    </ns2:parameter>
  </ns2:parameters>
  </ns2:property>
</ns2:properties>
```

4.7.4. Creating an update message

The update can be created in many ways but the most common approach is to build the update while modifications take place or to create one as the result of comparing old and new versions. It appears that comparing XML for differences is difficult. However, we can take advantage of the structure of calendaring entities to simplify the process. There are implementations available which take the diff approach to producing an update stream.

There are some special cases to consider when comparing. Some properties are multi-valued and may themselves appear more than once. There is no semantic information implied by any grouping though parameters may need to be taken into account. These properties need to be normalized before comparison and when updating them we produce a change which treats each value as a single property.

These properties are:

— `categories`
— `exdate`
— `freebusy`
— `rdate`

This normalization can take place before comparison.

Some properties are multi-valued and may only appear once. At the moment the only standard property is resource which may take a comma separated list. This should be treated as a single multi-valued property when comparing. The order is unimportant. Sorting the values may help.

Some properties may appear multiple times, for example comment. Comparison should take account of parameters. Ordering all properties appropriately allows for relatively simple comparison.

4.8. Deletion of resources

Deletion of calendar object resources is carried out by using a CalWS-SOAP deleteItem request with an `href` specifying the entity to be deleted. The `deleteItem` request is not valid when the `href` specifies a collection.

Table 42 — `DeleteItemType` fields

Field	Type	#	?	Description
href	string	1	Y	Identify the target of the request.

The service will respond with a `DeleteItemResponseType` containing the status and a possible error response. There are no additional elements.

:2013

4.8.1. Example — successful `deleteItem`

>>Request

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:deleteItem xmlns:ns2="http://docs.oasis-open.org/ws-calendar/ns/soap"
                    xmlns:ns3="urn:ietf:params:xml:ns:icalendar-2.0">
      <ns2:href>/user/douglm/calendar/1302620814655.ics</ns2:href>
    </ns2:deleteItem>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

>>Response

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:deleteItemResponse xmlns:ns2="http://docs.oasis-open.org/ws-calendar/ns/soap"
                            xmlns:ns3="urn:ietf:params:xml:ns:icalendar-2.0">
      <ns2:status>OK</ns2:status>
    </ns2:deleteItemResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

4.8.2. Example — unsuccessful `deleteItem`

>>Request

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:deleteItem xmlns:ns2="http://docs.oasis-open.org/ws-calendar/ns/soap"
                    xmlns:ns3="urn:ietf:params:xml:ns:icalendar-2.0">
      <ns2:href>/user/douglm/calendar/nosuchevent.ics</ns2:href>
    </ns2:deleteItem>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

>>Response

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:deleteItemResponse xmlns:ns2="http://docs.oasis-open.org/ws-calendar/ns/soap"
                            xmlns:ns3="urn:ietf:params:xml:ns:icalendar-2.0">
      <ns2:status>Error</ns2:status>
      <ns2:errorResponse>
        <ns2:targetDoesNotExist/>
      </ns2:errorResponse>
    </ns2:deleteItemResponse>
  </SOAP-ENV:Body>
```

```
</SOAP-ENV:Envelope>
```

4.9. Querying calendar resources

Querying provides a mechanism by which information can be obtained from the service through possibly complex queries. A skeleton iCalendar entity can be provided 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.

4.9.1. Calendar Query common types

The UTCTimeRangeType is used in a number of places to define a time range within which components must appear or property values must lie. The values are UTC time-date, the start is inclusive and the end is exclusive.

Table 43 — UTCTimeRangeType elements

Field	Type	#	?	Description
start	UTC date-time	1	Y	UTC inclusive start
end	UTC date-time	1	Y	UTC exclusive end

The TextMatchType is used to match text values in properties and parameters. The collation attribute species a collation as defined in [IETF RFC 4790](#).

Servers are REQUIRED to support the "i;ascii-casemap" and "i;octet" collations which provide a basic case insensitive and case sensitive match respectively.

Elements of this type take a string value which is matched according to the attributes.

Table 44 — TextMatchType attributes

Field	Type	#	?	Description
#collation	string	0..1	N	Collation name from [IETF RFC 4790](#).
#negate-condition	boolean	0..1	N	if "true" negates the condition

4.9.2. CompFilterType

This type defines a search query for the calendar query operation. It specifies the component types to return, absence tests or basic matching operations on properties and time ranges.

The top level comp-filter element (which must match a vCalendar component may contain zero or more comp-filter elements to match events, tasks or other contained components. These in turn may contain further nested comp-filter elements to match further levels of nested components.

Each may also contain prop-filter elements to test for the absence of properties or to match values. Only logical conjunctions are supported, that is, all elements of a comp-filter must match for the expression to match.

Table 45 — CompFilterType elements

Field	Type	#	?	Description
anyComp	AnyCompType	0..1	C	One of anyComp, vCalendar or a BaseComponentType must be supplied. anyComp indicates that any component will match.
xcal:vcalendar	xcal:VcalendarType	0..1	C	Matches vCalendar at the top level. Must be provided

Field	Type	#	?	Description
xcal: baseComponent	xcal: BaseComponentType	0..1	C	May be vEvent or vTODO for example.
#test	string	0..1	N	"anyof" is a logical OR of the child elements.
				"allof" is a logical AND of the child elements.
is-not-defined	empty	0..1	N	Only this element or one or more of time-range, prop-filter or comp-filter may be present
time-range	UTCTimeRangeType	0..1	N	
comp-filter	CompFilterType	1	Y	Match against contained components
prop-filter	PropFilterType	0..n	N	Match against component properties

4.9.3. PropFilterType

The prop-filter element may test for the absence of a property or match values or specify zero or more ParamFilterType elements to match against parameters.

Only logical conjunctions are supported, that is, all elements must match for the full expression to match.

Table 46 — PropFilterType elements

Field	Type	#	?	Description
xcal: baseProperty	xcal: BasePropertyType	1	Y	Specifies the property to be matched.
#test	string	0..1	N	"anyof" is a logical OR of the child elements.
				"allof" is a logical AND of the child elements.
is-not-defined	empty	0..1	N	Only this element or optionally one of time-range or text-match followed by param-filter
time-range	UTCTimeRangeType	0..1	N	
text-match	TextMatchtype	0..1	N	
param-filter	ParamFilterType	0..n	N	Match against property parameters

4.9.4. ParamFilterType

The ParamFilterType element may test for the absence of a parameter or match a value.

Table 47 — ParamFilterType elements

Field	Type	#	?	Description
xcal:baseParameter	xcal:BaseParameterType	1	Y	Specifies the parameter to be matched.
is-not-defined	empty	0..1	N	Only this element or text-match
text-match	TextMatchtype	0..1	N	

4.9.5. CalendarQueryType elements

Table 48 — CalendarQueryType elements

Field	Type	#	?	Description
href	string	1	Y	Identify the target of the request. "/" for the service.
allprop	empty	0..1	N	If present specifies all properties should be returned.
				One or none of allprop or iCalendar.
xcal:icalendar	xcal:IcalendarType	0..1	N	If present is a valueless iCalendar skeleton entity defining which

Field	Type	#	?	Description
				components and properties should be returned. If present `allprop` must NOT be present.
expand	ExpandType	0..1	N	A subclass of `UTCTimeRangeType`.
				Either expand or `limitRecurrenceSet` may be specified but not both.
				If specified recurring events are expanded and limited to the supplied time-range. All events times are converted to UTC.
				This option allows for simplified event handling for certain classes of client.
limitRecurrenceSet	LimitRecurrenceSetType	0..1	N	A subclass of `UTCTimeRangeType`.
				Either expand or `limitRecurrenceSet` may be specified but not both.
				If specified only overrides that fall within the specified time-range are returned.
				This helps to limit the size of the result-set when there are many overrides.
depth	string	0..1	N	Species depth for query. "1" ⇒ just targeted collection, "infinity" ⇒ query targeted and all sub-collections.
filter	FilterType	1	Y	Defines the search filter
/comp-filter	CompFilterType	1	Y	Defines the top-level component

4.9.6. Specifying data to be returned

This is achieved by specifying one of the following

— `allprop`: return all properties and calendar data. (some properties are specified as not being part of the `allprop` set so are not returned)
— Set the iCalendar element. This is an iCalendar valueless pattern entity which provides a map of the components and properties to be returned. Neither the pattern nor the returned result need to be valid iCalendar entities in that required properties may be absent if unselected.

4.9.7. Pre/postconditions for calendar queries

The preconditions as defined in [IETF RFC 4791, Section 7.8](#) apply here. CalWS errors may be reported by the service when preconditions or postconditions are violated.

4.9.8. Time range limited queries

Time-range limited retrieval has some special characteristics. The simplest case is a single event or task which overlaps the requested time-period. Recurring items and other components such as alarms complicate the picture.

4.9.9. 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 <<

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:calendarQuery xmlns:ns2="http://docs.oasis-open.org/ws-calendar/ns/soap"
                       xmlns:ns3="urn:ietf:params:xml:ns:icalendar-2.0">
      <ns2:href>/user/douglm/calendar</ns2:href>
      <ns3:icalendar>
        <ns3:vcalendar>
          <ns3:components>
            <ns3:vevent>
              <ns3:properties>
                <ns3:summary/>
                <ns3:dtstart/>
                <ns3:dtend/>
                <ns3:duration/>
                <ns3:uid/>
                <ns3:recurrence-id/>
                <ns3:rrule/>
                <ns3:rdate/>
                <ns3:exdate/>
              </ns3:properties>
            </ns3:vevent>
          </ns3:components>
        </ns3:vcalendar>
      </ns3:icalendar>
      <ns2:filter>
        <ns2:compFilter test="anyof">
          <ns3:vcalendar />
          <ns2:compFilter>
            <ns3:vevent />
            <ns2:time-range end="20110430T040000Z" start="20110401T040000Z"/>
          </ns2:compFilter>
        </ns2:filter>
    </ns2:calendarQuery>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

>> Response <<

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:calendarQueryResponse
              xmlns:ns2="http://docs.oasis-open.org/ws-calendar/ns/soap"
              xmlns:ns3="urn:ietf:params:xml:ns:icalendar-2.0">
      <ns2:status>OK</ns2:status>
      <ns2:response>
        <ns2:href>/user/douglm/calendar/1302105461170.ics</ns2:href>
        <ns2:changeToken>"20110406T155741Z-0"</ns2:changeToken>
        <ns2:propstat>
          <ns2:prop>
            <ns2:calendar-data content-type="application/xml+calendar" version=
"2.0">
              <ns3:icalendar>
                <ns3:vcalendar>
                  <ns3:properties>
```

```
              <ns3:prodid>
                <ns3:text>//Bedework.org//BedeWork V3.7//EN</ns3:text>
              </ns3:prodid>
              <ns3:version>
                <ns3:text>2.0</ns3:text>
              </ns3:version>
            </ns3:properties>
            <ns3:components>
              <ns3:vevent>
                <ns3:properties>
                  <ns3:dtend>
                    <ns3:date-time>20110406T160000Z</ns3:date-time>
                  </ns3:dtend>
                  <ns3:dtstart>
                    <ns3:date-time>20110406T150000Z</ns3:date-time>
                  </ns3:dtstart>
                  <ns3:summary>
                    <ns3:text>try this</ns3:text>
                  </ns3:summary>
                  <ns3:uid>
                    <ns3:text>1302105461170</ns3:text>
                  </ns3:uid>
                </ns3:properties>
              </ns3:vevent>
            </ns3:components>
          </ns3:vcalendar>
        </ns3:icalendar>
      </ns2:calendar-data>
    </ns2:prop>
    <ns2:status>OK</ns2:status>
  </ns2:propstat>
</ns2:response>
<ns2:response>
  <ns2:href>/user/douglm/calendar/CAL-00f1fc61-2f021bca-012f-022947f8-
00000006.ics</ns2:href>
    <ns2:changeToken>"20110405T140920Z-0"</ns2:changeToken>
    <ns2:propstat>
      <ns2:prop>
        <ns2:calendar-data content-type="application/xml+calendar" version=
"2.0">
          <ns3:icalendar>
            <ns3:vcalendar>
              <ns3:properties>
                <ns3:prodid>
                  <ns3:text>//Bedework.org//BedeWork V3.7//EN</ns3:text>
                </ns3:prodid>
                <ns3:version>
                  <ns3:text>2.0</ns3:text>
                </ns3:version>
              </ns3:properties>
              <ns3:components>
                <ns3:vevent>
                  <ns3:properties>
                    <ns3:duration>
                      <ns3:duration>PT1H</ns3:duration>
                    </ns3:duration>
                    <ns3:dtstart>
                      <ns3:parameters>
                        <ns3:tzid>
                          <ns3:text>America/New_York</ns3:text>
```

```
                              </ns3:tzid>
                            </ns3:parameters>
                            <ns3:date-time>20110412T110000</ns3:date-time>
                          </ns3:dtstart>
                          <ns3:summary>
                            <ns3:text>Test recurring event</ns3:text>
                          </ns3:summary>
                          <ns3:uid>
                            <ns3:text>CAL-00f1fc61-2f021bca-012f-022947f8-
00000006demobedework@mysite.edu</ns3:text>
                          </ns3:uid>
                          <ns3:rrule>
                            <ns3:recur>
                              <ns3:freq>WEEKLY</ns3:freq>
                              <ns3:count>2</ns3:count>
                              <ns3:interval>1</ns3:interval>
                            </ns3:recur>
                          </ns3:rrule>
                        </ns3:properties>
                      </ns3:vevent>
                      <ns3:vevent>
                        <ns3:properties>
                          <ns3:recurrence-id>
                            <ns3:parameters>
                              <ns3:tzid>
                                <ns3:text>America/New_York</ns3:text>
                              </ns3:tzid>
                            </ns3:parameters>
                            <ns3:date-time>20110419T150000Z</ns3:date-time>
                          </ns3:recurrence-id>
                          <ns3:duration>
                            <ns3:duration>PT1H</ns3:duration>
                          </ns3:duration>
                          <ns3:dtstart>
                            <ns3:parameters>
                              <ns3:tzid>
                                <ns3:text>America/New_York</ns3:text>
                              </ns3:tzid>
                            </ns3:parameters>
                            <ns3:date-time>20110419T120000</ns3:date-time>
                          </ns3:dtstart>
                          <ns3:summary>
                            <ns3:text>Test recurring event</ns3:text>
                          </ns3:summary>
                          <ns3:uid>
                            <ns3:text>CAL-00f1fc61-2f021bca-012f-022947f8-
00000006demobedework@mysite.edu</ns3:text>
                          </ns3:uid>
                        </ns3:properties>
                      </ns3:vevent>
                    </ns3:components>
                  </ns3:vcalendar>
                </ns3:icalendar>
              </ns2:calendar-data>
            </ns2:prop>
            <ns2:status>OK</ns2:status>
          </ns2:propstat>
        </ns2:response>
      </ns2:calendarQueryResponse>
    </SOAP-ENV:Body>
```

```
</SOAP-ENV:Envelope>
```

4.9.10. Free-busy queries

Freebusy queries are used to obtain `freebusy` information for a principal. The result contains information only for events to which the current principal has sufficient access and may be affected by components and rules available only to the server (for instance office hours availability).

These queries are carried out by using a CalWS-SOAP `freebusyReport` request with an `href` specifying a principal. The `freebusyReport` request is not valid when the `href` specifies any entity other than a principal. The query follows the specification defined in [CC/S 0903:2009](#) 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 users.

Freebusy information is returned by default as `xcalendar` `vfreebusy` components, as defined by [IETF RFC 6321](#). Such a component is not meant to conform to the requirements of VFREEBUSY components in [IETF RFC 5546](#). The VFREEBUSY component SHOULD conform to section [IETF RFC 5545, Section 4.6.4](#). 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.

4.9.11. Element values

Three values are provided: `href`; `start`; end. Only the `href` is required. The start and end are in XML UTC date/time format and are interpreted as follows:

4.9.11.1. start

Default	If omitted 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	An XML UTC date-time

EXAMPLE

```
2011-12-01T10:15:00Z
```

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

4.9.11.2. 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`

:2013

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 response as a server may return a different time range than the requested range.

4.9.12. Examples

The following is an unsuccessful request targeting an invalid resource.

```
>> Request <<

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:freebusyReport
          xmlns:ns2="http://docs.oasis-open.org/ws-calendar/ns/soap"
          xmlns:ns3="urn:ietf:params:xml:ns:icalendar-2.0">
      <ns2:href>/user/douglm/calendar</ns2:href>
      <ns2:time-range>
        <ns2:start>2011-04-01T04:00:00Z</ns2:start>
        <ns2:end>2011-04-30T04:00:00Z</ns2:end>
      </ns2:time-range>
    </ns2:freebusyReport>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

>> Response <<

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:freebusyReportResponse
          xmlns:ns2="http://docs.oasis-open.org/ws-calendar/ns/soap"
          xmlns:ns3="urn:ietf:params:xml:ns:icalendar-2.0">
      <ns2:status>Error</ns2:status>
      <ns2:message>Only principal href supported</ns2:message>
    </ns2:freebusyReportResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The following is an example of a request to retrieve Freebusy data for a user:

```
>> Request <<

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:freebusyReport
          xmlns:ns2="http://docs.oasis-open.org/ws-calendar/ns/soap"
          xmlns:ns3="urn:ietf:params:xml:ns:icalendar-2.0">
      <ns2:href>/principals/users/douglm</ns2:href>
      <ns2:time-range>
        <ns2:start>2011-04-01T04:00:00Z</ns2:start>
        <ns2:end>2011-04-30T04:00:00Z</ns2:end>
      </ns2:time-range>
    </ns2:freebusyReport>
  </SOAP-ENV:Body>
```

```
</SOAP-ENV:Envelope>

>> Response <<

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:freebusyReportResponse
          xmlns:ns2="http://docs.oasis-open.org/ws-calendar/ns/soap"
          xmlns:ns3="urn:ietf:params:xml:ns:icalendar-2.0">
      <ns2:status>OK</ns2:status>
      <ns3:icalendar>
        <ns3:vcalendar>
          <ns3:properties>
            <ns3:prodid>
              <ns3:text>//Bedework.org//BedeWork V3.7//EN</ns3:text>
            </ns3:prodid>
            <ns3:version>
              <ns3:text>2.0</ns3:text>
            </ns3:version>
          </ns3:properties>
          <ns3:components>
            <ns3:vfreebusy>
              <ns3:properties>
                <ns3:attendee>
                  <ns3:parameters>
                    <ns3:partstat>
                      <ns3:text>NEEDS-ACTION</ns3:text>
                    </ns3:partstat>
                  </ns3:parameters>
                  <ns3:cal-address>mailto:douglm@mysite.edu</ns3:cal-address>
                </ns3:attendee>
                <ns3:created>
                  <ns3:utc-date-time>2011-06-30T15:45:56Z</ns3:utc-date-time>
                </ns3:created>
                <ns3:dtend>
                  <ns3:date-time>2011-04-30T00:00:00Z</ns3:date-time>
                </ns3:dtend>
                <ns3:dtstamp>
                  <ns3:utc-date-time>2011-06-30T15:45:56Z</ns3:utc-date-time>
                </ns3:dtstamp>
                <ns3:dtstart>
                  <ns3:date-time>2011-04-01T00:00:00Z</ns3:date-time>
                </ns3:dtstart>
                <ns3:freebusy>
                  <ns3:parameters>
                    <ns3:fbtype>
                      <ns3:text>BUSY</ns3:text>
                    </ns3:fbtype>
                  </ns3:parameters>
                  <ns3:period>
                    <ns3:start>2011-04-06T15:00:00Z</ns3:start>
                    <ns3:end>2011-04-06T16:00:00Z</ns3:end>
                  </ns3:period>
                </ns3:freebusy>
                <ns3:last-modified>
                  <ns3:utc-date-time>2011-06-30T15:45:56Z</ns3:utc-date-time>
                </ns3:last-modified>
                <ns3:organizer>
```

```
                    <ns3:parameters/>
                    <ns3:cal-address>mailto:douglm@mysite.edu</ns3:cal-address>
                 </ns3:organizer>
                 <ns3:uid>
                    <ns3:text>2UTDVPZ9H0EQL9QISI44SP5IFPC4N75</ns3:text>
                 </ns3:uid>
               </ns3:properties>
            </ns3:vfreebusy>
          </ns3:components>
        </ns3:vcalendar>
      </ns3:icalendar>
    </ns2:freebusyReportResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

4.10. Multiple operations

Each of the previously described operations acts upon a single entity or resource only. Frequently we have the need to update an interconnected set of entities so that we maintain the consistency of the structure. This requires an atomic operation which can successfully update all the entities or roll back the operation on failure.

The `MultiOpType` operation provides such a feature. It is essentially a wrapper around any of the other operations which guarantees the success of the entire set or a roll back. Using the id attribute for requests, each individual response can be located in the result. The `MultiOpType` request takes the following elements

Table 49 — `MultiOpType` elements

Field	Type	#	?	Description
operations	Sequence of BaseOperationType	1	Y	Contains one or more operations

The response type is also simple containing a single element containing all the responses.

Table 50 — `MultiOpResponseType` elements

Field	Type	#	?	Description
responses	Sequence of BaseResponseType	1	Y	Contains zero or more responses

5. Conformance

Certain calendaring properties and components are interrelated and it is necessary to have knowledge of all these properties and their current values to allow consistent update and understanding of a target component. The normative definition for these relationships is RFC 5445, RFC 5446 and related RFCs.

However, those specifications assume a complete view of entities being fetched or updated. This specification allows updates of entities when only a partial view is available. In fact it is the very nature of SOAP based transaction to provide such a partial view. Given that, parties attempting to update entities MUST have sufficient information to ensure the end result is consistent. Services allowing updates to entities MUST ensure that the result after an update operation is still internally consistent.

5.1. Start, end and duration in calendar components

A period of time is fully specified by a start and an end or duration.

5.1.1. Updating, transporting and maintaining start, and and duration

—   For all components the calculated or specified start must be at or before the end.

— When a system updates or stores a calendar component it MUST retain the relationship of start, end and duration. Applications MUST NOT without good cause, change a start and end pair into a start and duration nor the reverse. Semantically they are not equivalent when DST transitions occur during the time of the event.
— For interoperability, iCalendar based systems SHOULD avoid the use of weekly durations and XML based systems SHOULD avoid the use of yearly durations.

5.1.2. VEVENT

— The three properties are DTSTART, DTEND and DURATION.
— DTSTART MUST appear once and only one of DTEND or DURATION MAY be present.
— The DTSTART property for a VEVENT specifies the inclusive start of the event. For recurring events, it also specifies the very first instance in the recurrence set.
— The DTEND property for a VEVENT calendar component specifies the non-inclusive end of the event.
— For cases where a VEVENT calendar component specifies a DTSTART property with a DATE value type but no DTEND nor DURATION property, the event's duration is taken to be one day.
— For cases where a VEVENT calendar component specifies a DTSTART property with a DATE-TIME value type but no DTEND nor DURATION property, the event ends on the same calendar date and time of day specified by the DTSTART property, that is, it signifies a zero length instant in time.

5.1.3. VTODO

— The three properties are DTSTART, DUE, DURATION.
— DTSTART MAY appear once.
— Either DUE or DURATION MAY appear in a VTODO, but DUE and DURATION MUST NOT occur in the same VTODO.
— If DURATION does appear in a VTODO, then DTSTART MUST also appear in the same VTODO.
— The three properties for a VTODO are related in the same way as for VEVENT. Additionally a VTODO calendar component without the DTSTART and DUE (or DURATION) properties specifies a VTODO that will be associated with each successive calendar date, until it is completed.

5.1.4. VJOURNAL

— DTSTART only, which may be a date or date-time value.

5.1.5. VAVAILABILITY

— DTSTART and DTEND if specified MUST be date-time values.
— DTSTART MAY appear once and signifies start of the busy period.
— Only one of DTEND or DURATION MAY appear and signify the end of the busy period.
— If DURATION does appear in a VAVAILABILITY, then DTSTART MUST also appear in the same VAVAILABILITY.

5.1.6. AVAILABILITY

— DTSTART and DTEND if specified MUST be date-time values.
— DTSTART MUST appear once and signifies start of the free period.
— Only one of DTEND or DURATION MAY appear and signify the end of the free period.

5.2. Recurrences

— The RECURRENCE-ID is a property of each instance of a recurring event. It is calculated from the DTSTART and the recurrence rules or added to the set by the RDATE property.
— RDATE, EXDATE and RECURRENCE-ID must take the same form as the DTSTART. That is if DTSTART is a DATE value then the RDATE and EXDATE must be DATE. If DTSTART is a date-time the RDATE and EXDATE values must take the same form, including the same timezone.

— Overrides to an instance are specified by completely specifying the instance with the appropriate RECURRENCE-ID property.
— An RDATE adds an instance to the recurrence set.
— An EXDATE deletes an instance by specifying the recurrence id(s) to be deleted. Applications SHOULD NOT specify overrides for instances so deleted.
— The recurrence set is calculated from the RRULE and RDATES and then applying any EXDATE properties. That is EXDATE takes precedence over RDATE and the RRULE.

5.3. Alarms

— Alarms are typically anchored to the start or end of an event or task. This is defined by the RELATED parameter to the TRIGGER property.

5.4. Unrecognized or unsupported elements

— A system SHOULD reject any attempt to store components which it does not support. A SYSTEM MUST advertise which components are supported through the use of the supportedCalendarComponentSet property.
— A system MUST ignore any elements it does not understand.

6. 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 WSCalendar 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

Bibliography

[1]   WS-Addr, W3C Recommendation,Web Services Addressing 1.0 — Core, and Web Services Addressing 1.0 — SOAP Binding, 9 May 2006 http://www.w3.org/2002/ws/addr/

[2]   WT-I-Basic, Basic Profile Version 1.1, 10 April 2006 http://www.ws-i.org/Profiles/BasicProfile-1.1-2006-04-10.html

[3]   WS-I-Bind, Web Services-Interoperability Organization (WS-I) Simple SOAP Binding Profile Version 1.0, 24 August 2004 http://www.ws-i.org/Profiles/SimpleSoapBindingProfile-1.0-2004-08-24.html