

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..

This document is copyright ©2012 by The Calendaring and Scheduling Consortium and is licensed under the Creative Commons 3.0 Unported License: <http://creativecommons.org/licenses/by/3.0/>.

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 (dougln@rpi.edu) Rensselaer Polytechnic Institute

Editor:

Michael Douglass (dougln@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	5
1.1	Terminology	5
1.2	Normative References	5
1.3	Non-Normative References	5
1.4	Namespace.....	5
2	Calendar Services	7
2.1	Overview of the protocol	7
2.1.1	Calendar Object Resources	7
2.1.2	Timezone information	7
2.1.3	Issues not addressed by this specification.	8
2.1.3.1	Access Control.....	8
2.1.3.2	Provisioning	8
2.1.3.3	Copy/Move.....	8
2.1.3.4	Creating Collections.....	8
2.1.3.5	Retrieving collections	8
2.1.3.6	Setting service and resource properties.....	8
2.1.4	CalWS Glossary	8
2.1.4.1	Hrefs	8
2.1.4.2	Calendar Object Resource.....	8
2.1.4.3	Calendar Collection.....	8
2.1.4.4	Scheduling Calendar Collection.....	9
2.1.4.5	Principal Home.....	9
2.2	Error conditions.....	9
2.2.1	Example: error with CalDAV error condition	9
3	Properties and link relations	10
3.1	Property and relation-type URIs	10
3.2	supported-features property.	10
3.3	max-attendees-per-instance.....	10
3.4	max-date-time.....	10
3.5	max-instances.....	10
3.6	max-resource-size	10
3.7	min-date-time.....	11
3.8	description.....	11
3.9	timezone-service relation.....	11
3.10	principal-home relation.	11
3.11	current-principal-freebusy relation.	11
3.12	principal-freebusy relation.....	11
3.13	child-collection relation.	11
3.14	created link property	12
3.15	last-modified property	12
3.16	displayname property	12
3.17	timezone property	12
3.18	owner property.....	12
3.19	collection link property	12

3.20	calendar-collection link property	12
3.21	calWS:privilege-set XML element.....	13
4	Retrieving Collection and Service Properties.....	14
4.1	Request parameters	14
4.2	Responses:	14
4.3	Example - retrieving server properties:.....	14
5	Creating Calendar Object Resources.....	16
5.1	Request parameters	16
5.2	Responses:	16
5.3	Preconditions for Calendar Object Creation	16
5.4	Example - successful POST:.....	17
5.5	Example - unsuccessful POST:	17
6	Retrieving resources.....	18
6.1	Request parameters	18
6.2	Responses:	18
6.3	Example - successful fetch:	18
6.4	Example - unsuccessful fetch:	18
7	Updating resources	19
7.1	Responses:	19
8	Deletion of resources.....	21
8.1	Delete for Collections.....	21
8.2	Responses:	21
9	Querying calendar resources	22
9.1	Limiting data returned	22
9.2	Pre/postconditions for calendar queries	22
9.3	Example: time range limited retrieval	22
10	Free-busy queries.....	27
10.1	ACCEPT header	27
10.2	URL Query Parameters	27
10.2.1	start.....	27
10.2.2	end.....	28
10.2.3	period.....	28
10.2.4	account.....	28
10.3	URL parameters - notes	28
10.4	HTTP Operations	28
10.5	Response Codes	28
10.6	Examples	29
11	Conformance.....	32
Appendix A.	Acknowledgments.....	33
Appendix B.	An Introduction to Internet Calendaring	34
B.1	icalendar	34
B.1.1	History	34
B.1.2	Data model.....	34
B.1.3	Scheduling	35
B.1.4	Extensibility	35

B.2 Calendar data access and exchange protocols	35
B.2.1 Internet Calendar Subscriptions.....	35
B.2.2 CalDAV	35
B.2.3 ActiveSync/SyncML	36
B.2.4 CalWS.....	36
B.2.5 iSchedule	36
B.3 References	36
Appendix C. Revision History	37

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

36 2 Calendar Services

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

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

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

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

49 2.1 Overview of the protocol

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

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

```
55 POST /user/fred/calendar/ HTTP/1.1  
56 ...  
57 X-HTTP-Method-Override: PUT  
58 Properties
```

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

62 The following operations are defined by this specification:

- 63 • Retrieval and update of service and resource properties
- 64 • Creation of a calendar object
- 65 • Retrieval of a calendar object
- 66 • Update of a calendar object
- 67 • Deletion of a calendar object
- 68 • Query
- 69 • Free-busy query

70 2.1.1 Calendar Object Resources

71 The same restrictions apply to Calendar Object Resources as specified in CalDAV [RFC4791] section
72 4.2. An additional constraint for CalWS is that no timezone specifications are transferred.

73 2.1.2 Timezone information

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

78 **2.1.3 Issues not addressed by this specification.**

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

81 **2.1.3.1 Access Control**

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

84 **2.1.3.2 Provisioning**

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

88 **2.1.3.3 Copy/Move**

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

93 **2.1.3.4 Creating Collections**

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

96 **2.1.3.5 Retrieving collections**

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

99 **2.1.3.6 Setting service and resource properties.**

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

102 **2.1.4 CalWS Glossary**

103 **2.1.4.1 Hrefs**

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

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

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

108 **2.1.4.2 Calendar Object Resource**

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

111 **2.1.4.3 Calendar Collection**

112 A folder only allowed to contain calendar object resources.

113 2.1.4.4 Scheduling Calendar Collection

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

116 2.1.4.5 Principal Home

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

119 2.2 Error conditions

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

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

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

129 2.2.1 Example: error with CalDAV error condition

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

139 3 Properties and link relations

140 3.1 Property and relation-type URIs

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

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

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

149 corresponds to

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

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

153 3.2 supported-features property.

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

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

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

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

162 3.3 max-attendees-per-instance

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

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

165 3.4 max-date-time

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

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

168 3.5 max-instances

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

170 Defines the maximum number of instances allowed per event or task

171 3.6 max-resource-size

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

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

175 **3.7 min-date-time**

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

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

179 **3.8 description**

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

181 Provides some descriptive text for the targeted collection.

182 **3.9 timezone-service relation.**

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

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

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

189 **3.10 principal-home relation.**

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

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

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

194 **3.11 current-principal-freebusy relation.**

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

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

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

199 **3.12 principal-freebusy relation.**

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

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

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

204 **3.13 child-collection relation.**

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

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

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

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

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

220 3.14 created link property

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

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

```
224 <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/created"
225 >1985-04-12T23:20:50.52Z</Property>
```

226 3.15 last-modified property

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

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

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

232 3.16 displayname property

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

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

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

238 3.17 timezone property

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

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

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

243 3.18 owner property

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

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

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

248 3.19 collection link property

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

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

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

254 3.20 calendar-collection link property

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

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

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

261 3.21 calWS:privilege-set XML element

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

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

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

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

- 268 • calWS: Read - current principal has read access
- 269 • calWS: Write - current principal has write access

```
270 <calWS:privilege-set>  
271 <calWS:privilege><calWS:read></calWS:privilege>  
272 <calWS:privilege><calWS:write></calWS:privilege>  
273 </calWS:privilege-set>
```

274

4 Retrieving Collection and Service Properties

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

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

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

281 The service properties define the global limits and defaults. Any properties defined on collections within
282 the service hierarchy override those service defaults. The service may choose to prevent such overriding
283 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:

```
291 >>Request
292
293 GET / HTTP/1.1
294 Host: example.com
295 ACCEPT:application/xrd+xml
296
297 >>Response
298 <XRD xmlns="http://docs.oasis-open.org/ns/xri/xrd-1.0"
299     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
300   <Expires>1970-01-01T00:00:00Z</Expires>
301   <Subject>http://example.com/calWS</Subject>
302   <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/created"
303     >1970-01-01</Property>
304
305   <Link rel=" http://docs.oasis-open.org/ws-calendar/ns/REST/timezone-
306 service"
307     href="http://example.com/tz" />
308
309   <calWS:privilege-set>
310   <calWS:privilege><calWS:read></calWS:privilege>
311   </calWS:privilege-set>
312
313   <Link rel=" http://docs.oasis-open.org/ws-calendar/ns/REST/principal-home"
314     type="collection"
315     href="http://example.com/calWS/user/fred">
316   <Title xml:lang="en">Fred's calendar home</Title>
317   </Link>
318
319   <Link rel=" http://docs.oasis-open.org/ws-calendar/ns/REST/child-
320 collection"
321     type="calendar,scheduling"
```

```
322         href="http://example.com/calWS/user/fred/calendar">
323     <Title xml:lang="en">Calendar</Title>
324     </Link>
325
326     <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/max-
327 instances"
328         >1000</Property>
329
330     <Property type=" http://docs.oasis-open.org/ws-calendar/ns/REST/max-
331 attendees-per-instance"
332         >100</Property>
333
334 </XRD>
335
```


336
337
338
339

340
341

342
343
344

345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376

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;

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

389 5.4 Example - successful POST:

```

390 >>Request
391
392 POST /user/fred/calendar/?action=create HTTP/1.1
393 Host: example.com
394 Content-Type: application/xml+calendar; charset="utf-8"
395 Content-Length: ?
396
397 <?xml version="1.0" encoding="utf-8" ?>
398 <icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
399   <vcalendar>
400     ...
401   </vcalendar>
402 </icalendar>
403
404 >>Response
405
406 HTTP/1.1 201 Created
407 Location: http://example.com/user/fred/calendar/event1.ics

```

408 5.5 Example - unsuccessful POST:

```

409 >>Request
410
411 POST /user/fred/readcalendar/?action=create HTTP/1.1
412 Host: example.com
413 Content-Type: text/text; charset="utf-8"
414 Content-Length: ?
415
416 This is not an xml calendar object
417
418 >>Response
419
420 HTTP/1.1 403 Forbidden
421 <?xml version="1.0" encoding="utf-8"
422   xmlns:D="DAV:"
423   xmlns:C="urn:ietf:params:xml:ns:caldav" ?>
424 <D:error>
425   <C:supported-calendar-data/>
426   <D:description>Not an icalendar object</D:description>
427 </D:error>

```

428 6 Retrieving resources

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

432 6.1 Request parameters

- 433 • none

434 6.2 Responses:

- 435 • 200: OK
- 436 • 403: Forbidden - no access
- 437 • 406 The requested format specified in the accept header is not supported.

438 6.3 Example - successful fetch:

```
439 >>Request
440
441 GET /user/fred/calendar/event1.ics HTTP/1.1
442 Host: example.com
443
444 >>Response
445
446 HTTP/1.1 200 OK
447 Content-Type: application/xml+calendar; charset="utf-8"
448 Content-Length: ?
449
450 <?xml version="1.0" encoding="utf-8" ?>
451 <icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
452   <vcalendar>
453     ...
454   </vcalendar>
455 </icalendar>
```

456 6.4 Example - unsuccessful fetch:

```
457 >>Request
458
459 PUT /user/fred/calendar/noevent1.ics HTTP/1.1
460 Host: example.com
461
462 >>Response
463
464 HTTP/1.1 404 Not found
```

465

7 Updating resources

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

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

470 Preconditions as specified in Section 5.3 are applicable.

471 7.1 Responses:

472 • 200: OK

473 • 304: Not modified - entity was modified by some other request

474 • 403: Forbidden - no access, does not exist etc. See error response

475

476 *Example 7-1: Successful Update*

```
477 >>Request
478
479 PUT /user/fred/calendar/event1.ics HTTP/1.1
480 Host: example.com
481 Content-Type: application/xml+calendar; charset="utf-8"
482 Content-Length: ?
483
484 <?xml version="1.0" encoding="utf-8" ?>
485 <icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
486   <vcalendar>
487     ...
488   </vcalendar>
489 </icalendar>
490
491 >>Response
492
493 HTTP/1.1 200 OK
```

494 *Example 7-2: Unsuccessful Update*

```
495 >>Request
496
497 PUT /user/fred/readcalendar/event1.ics HTTP/1.1
498 Host: example.com
499 Content-Type: application/xml+calendar; charset="utf-8"
500 Content-Length: ?
501
502 <?xml version="1.0" encoding="utf-8" ?>
503 <icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
504   <vcalendar>
505     ...
506   </vcalendar>
507 </icalendar>
508
509 >>Response
510
511 HTTP/1.1 403 Forbidden
512 Content-Type: application/xml; charset="utf-8"
513 Content-Length: xxxx
514
515 <?xml version="1.0" encoding="utf-8"
```

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

522 8 Deletion of resources

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

527 8.1 Delete for Collections

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

531 8.2 Responses:

- 532 • 200: OK
- 533 • 403: Forbidden - no access
- 534 • 404: Not Found

535 9 Querying calendar resources

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

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

542 The current specification uses CalDAV multiget and calendar-query XML bodies as specified in [RFC
543 4791] with certain limitations and differences.

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

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

551 9.1 Limiting data returned

552 This is achieved by specifying one of the following

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

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

559 9.2 Pre/postconditions for calendar queries

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

562 9.3 Example: time range limited retrieval

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

```
565 >> Request <<
566
567 POST /user/fred/calendar/ HTTP/1.1
568 Host: calWS.example.com
569 Depth: 1
570 Content-Type: application/xml; charset="utf-8"
571 Content-Length: xxxx
572
573 <?xml version="1.0" encoding="utf-8" ?>
574 <C:calendar-query xmlns:D="DAV:"
575                 xmlns:C="urn:ietf:params:xml:ns:caldav">
576   <D:prop>
577     <D:getetag/>
578   <C:calendar-data content-type="application/xml+calendar" >
```

```

579     <C:comp name="VCALENDAR">
580       <C:prop name="VERSION"/>
581       <C:comp name="VEVENT">
582         <C:prop name="SUMMARY"/>
583         <C:prop name="UID"/>
584         <C:prop name="DTSTART"/>
585         <C:prop name="DTEND"/>
586         <C:prop name="DURATION"/>
587         <C:prop name="RRULE"/>
588         <C:prop name="RDATE"/>
589         <C:prop name="EXRULE"/>
590         <C:prop name="EXDATE"/>
591         <C:prop name="RECURRENCE-ID"/>
592       </C:comp>
593     </C:comp>
594   </C:calendar-data>
595 </D:prop>
596 <C:filter>
597   <C:comp-filter name="VCALENDAR">
598     <C:comp-filter name="VEVENT">
599       <C:time-range start="20060104T000000Z"
600         end="20060105T000000Z"/>
601     </C:comp-filter>
602   </C:comp-filter>
603 </C:filter>
604 </C:calendar-query>
605
606 >> Response <<
607
608 HTTP/1.1 207 Multi-Status
609 Date: Sat, 11 Nov 2006 09:32:12 GMT
610 Content-Type: application/xml; charset="utf-8"
611 Content-Length: xxxx
612
613 <?xml version="1.0" encoding="utf-8" ?>
614 <D:multistatus xmlns:D="DAV:"
615   xmlns:C="urn:ietf:params:xml:ns:caldav">
616   <D:response>
617     <D:href>http://cal.example.com/bernard/work/abcd2.ics</D:href>
618     <D:propstat>
619       <D:prop>
620         <D:getetag>"fffff-abcd2"</D:getetag>
621         <C:calendar-data content-type="application/xml+calendar" >
622           <xc:icalendar
623             xmlns:xc="urn:ietf:params:xml:ns:icalendar-2.0">
624             <xc:vcalendar>
625               <xc:properties>
626                 <xc:calscale><text>GREGORIAN</text></xc:calscale>
627                 <xc:prodid>
628                   <xc:text>-//Example Inc.//Example Calendar//EN</xc:text>
629                 </xc:prodid>
630                 <xc:version><xc:text>2.0</xc:text></xc:version>
631               </xc:properties>
632               <xc:components>
633                 <xc:vevent>
634                   <xc:properties>
635                     <xc:dtstart>
636                       <xc:parameters>
637                         <xc:tzid>US/Eastern</xc:tzid>
638                       <xc:parameters>
639                         <xc:date-time>20060102T120000</xc:date-time>
640                     </xc:dtstart>
641                     <xc:duration><xc:duration>PT1H</xc:duration></xc:duration>
642                   <xc:summary>

```



```

643     <xc:text>Event #2</xc:text>
644 </xc:summary>
645 <xc:uid>
646   <xc:text>00959BC664CA650E933C892C@example.com</xc:text>
647 </xc:uid>
648 <xc:rrule>
649   <xc:recur>
650     <xc:freq>DAILY</xc:freq>
651     <xc:count>5</xc:count>
652   </xc:recur>
653 </xc:rrule>
654 </xc:properties>
655 </xc:vevent>
656
657 <xc:vevent>
658   <xc:properties>
659     <xc:dtstart>
660       <xc:parameters>
661         <xc:tzid>US/Eastern<xc:tzid>
662       </xc:parameters>
663       <xc:date-time>20060104T140000</xc:date-time>
664     </xc:dtstart>
665     <xc:duration><xc:duration>PT1H</xc:duration></xc:duration>
666     <xc:summary>
667       <xc:text>Event #2 bis</xc:text>
668     </xc:summary>
669     <xc:uid>
670       <xc:text>00959BC664CA650E933C892C@example.com</xc:text>
671     </xc:uid>
672     <xc:recurrence-id>
673       <xc:parameters>
674         <xc:tzid>US/Eastern<xc:tzid>
675       </xc:parameters>
676       <xc:date-time>20060104T120000</xc:date-time>
677     </xc:recurrence-id>
678     <xc:rrule>
679       <xc:recur>
680         <xc:freq>DAILY</xc:freq>
681         <xc:count>5</xc:count>
682       </xc:recur>
683     </xc:rrule>
684   </xc:properties>
685 </xc:vevent>
686
687 <xc:vevent>
688   <xc:properties>
689     <xc:dtstart>
690       <xc:parameters>
691         <xc:tzid>US/Eastern<xc:tzid>
692       </xc:parameters>
693       <xc:date-time>20060106T140000</xc:date-time>
694     </xc:dtstart>
695     <xc:duration><xc:duration>PT1H</xc:duration></xc:duration>
696     <xc:summary>
697       <xc:text>Event #2 bis bis</xc:text>
698     </xc:summary>
699     <xc:uid>
700       <xc:text>00959BC664CA650E933C892C@example.com</xc:text>
701     </xc:uid>
702     <xc:recurrence-id>
703       <xc:parameters>
704         <xc:tzid>US/Eastern<xc:tzid>
705       </xc:parameters>
706       <xc:date-time>20060106T120000</xc:date-time>

```

```

707     </xc:recurrence-id>
708     <xc:rrule>
709         <xc:recur>
710             <xc:freq>DAILY</xc:freq>
711             <xc:count>5</xc:count>
712         </xc:recur>
713     </xc:rrule>
714 </xc:properties>
715 </xc:vevent>
716 </xc:components>
717 </xc:vcalendar>
718 </xc:icalendar>
719     </C:calendar-data>
720 </D:prop>
721     <D:status>HTTP/1.1 200 OK</D:status>
722 </D:propstat>
723 </D:response>
724 <D:response>
725     <D:href>http://cal.example.com/bernard/work/abcd3.ics</D:href>
726     <D:propstat>
727         <D:prop>
728             <D:getetag>"fffff-abcd3"</D:getetag>
729             <C:calendar-data content-type="application/xml+calendar" >
730                 <xcal:icalendar
731                     xmlns:xc="urn:ietf:params:xml:ns:icalendar-2.0">
732 <xc:vcalendar>
733 <xc:properties>
734 <xc:calscale><text>GREGORIAN</text></xc:calscale>
735 <xc:prodid>
736 <xc:text>-//Example Inc.//Example Calendar//EN</xc:text>
737 </xc:prodid>
738 <xc:version><xc:text>2.0</xc:text></xc:version>
739 </xc:properties>
740 <xc:components>
741 <xc:vevent>
742 <xc:properties>
743 <xc:dtstart>
744 <xc:parameters>
745 <xc:tzid>US/Eastern<xc:tzid>
746 <xc:parameters>
747 <xc:date-time>20060104T100000</xc:date-time>
748 </xc:dtstart>
749 <xc:duration><xc:duration>PT1H</xc:duration></xc:duration>
750 <xc:summary>
751 <xc:text>Event #3</xc:text>
752 </xc:summary>
753 <xc:uid>
754 <xc:text>DC6C50A017428C5216A2F1CD@example.com</xc:text>
755 </xc:uid>
756 <xc:rrule>
757 <xc:recur>
758 <xc:freq>DAILY</xc:freq>
759 <xc:count>5</xc:count>
760 </xc:recur>
761 </xc:rrule>
762 </xc:properties>
763 </xc:vevent>
764 </xc:components>
765 </xc:vcalendar>
766 </xc:icalendar>
767     </C:calendar-data>
768 </D:prop>
769     <D:status>HTTP/1.1 200 OK</D:status>
770 </D:propstat>

```

771
772

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

773 10 Free-busy queries

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

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

779 The possible targets are:

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

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

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

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

792 10.1 ACCEPT header

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

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

796 10.2 URL Query Parameters

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

799 10.2.1 start

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

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

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

```
806 2007-01-02T13:00:00-08:00
```

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

808 **Example:**

```
809 2007-02-03T15:30:00-0800
```

```
810 2007-12-01T10:15:00Z
```

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

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

815 10.2.2 end

816 **Default:** Same as start

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

818 **Format:** Same as start

819 **Example:** Same as start

820 10.2.3 period

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

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

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

825 **Example:**

826 P42D

827 10.2.4 account

828 **Default:** none

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

831 **Format:** Server specific

832 **Example:**

833 fred
834 /principals/users/jim
835 user1@example.com

836 10.3 URL parameters - notes

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

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

841 10.4 HTTP Operations

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

845 10.5 Response Codes

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

- 848 • 200 OK
- 849 • 302 Found
- 850 • 400 Start parameter could not be understood / End parameter could not be understood / Period
851 parameter could not be understood
- 852 • 401 Unauthorized
- 853 • 403 Forbidden
- 854 • 404 The data for the requested principal is not currently available, but may be available later.

- 855 • 406 The requested format in the accept header is not supported.
- 856 • 410 The data for the requested principal is no longer available
- 857 • 500 General server error

858 10.6 Examples

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

```
860 http://www.example.com/freebusy/user1@example.com?
861 start=2007-09-01T00:00:00-08:00&end=2007-09-31T00:00:00-08:00
862
863 http://www.example.com/freebusy/user1@example.com?
864 start=2007-09-01T00:00:00-08:00&end=2007-09-31T00:00:00-08:00
865
866 http://www.example.com/freebusy/user1@example.com
867
868 http://www.example.com/freebusy?user=user%201@example.com&
869 start=2008-01-01T00:00:00Z&end=2008-12-31T00:00:00Z
```

870 Some Request/Response Examples:

871 A URL with no query parameters:

```
872 >> Request <<
873 GET /freebusy/bernard/ HTTP/1.1
874 Host: www.example.com
875
876 >> Response <<
877 HTTP/1.1 200 OK
878 Content-Type: application/xml+calendar; charset="utf-8"
879 Content-Length: xxxx
880
881 <xc:icalendar xmlns:xc="urn:ietf:params:xml:ns:icalendar-2.0">
882   <xc:vcalendar>
883     <xc:properties>
884       <xc:calscale><text>GREGORIAN</text></xc:calscale>
885       <xc:prodid>
886         <xc:text>-//Example Inc.//Example Calendar//EN</xc:text>
887       </xc:prodid>
888       <xc:version><xc:text>2.0</xc:text></xc:version>
889     </xc:properties>
890     <xc:components>
891       <xc:vfreebusy>
892         <xc:properties>
893           <xc:uid>
894             <xc:text>76ef34-54a3d2@example.com</xc:text>
895           </xc:uid>
896           <xc:dtstart>
897             <xc:date-time>20060101T000000Z</xc:date-time>
898           </xc:dtstart>
899           <xc:dtend>
900             <xc:date-time>20060108T000000Z</xc:date-time>
901           </xc:dtend>
902           <xc:dtstamp>
903             <xc:date-time>20050530T123421Z</xc:date-time>
904           </xc:dtstamp>
905           <xc:freebusy>
906             <xc:parameters>
907               <xc:fctype>BUSY TENTATIVE</xc:fctype>
908             </xc:parameters>
909             <xc:period>20060102T100000Z/20060102T120000Z</xc:period>
910           </xc:freebusy>
911           <xc:freebusy>
912             <xc:period>20060103T100000Z/20060103T120000Z</xc:period>
```

```

913     </xc:freebusy>
914     <xc:freebusy>
915         <xc:period>20060104T100000Z/20060104T120000Z</xc:period>
916     </xc:freebusy>
917     <xc:freebusy>
918         <xc:parameters>
919             <xc:fbtype>BUSYUNAVAILABLE<xc:fbtype>
920         <xc:parameters>
921         <xc:period>20060105T100000Z/20060105T120000Z</xc:period>
922     </xc:freebusy>
923     <xc:freebusy>
924         <xc:period>20060106T100000Z/20060106T120000Z</xc:period>
925     </xc:freebusy>
926 </xc:vfreebusy>
927 </xc:components>
928 </xc:vcalendar>
929 <xc:icalendar>

```

930 A URL with start and end parameters:

```

931 >> Request <<
932 GET /freebusy/user1@example.com?start=2007-09-01T00:00:00-08:00&end=2007-09-
933 31T00:00:00-08:00
934 HTTP/1.1
935 Host: www.example.com
936
937 >> Response <<
938 HTTP/1.1 200 OK
939 Content-Type: application/xml+calendar; charset="utf-8"
940 Content-Length: xxxx
941
942 <xc:icalendar xmlns:xc="urn:ietf:params:xml:ns:icalendar-2.0">
943     <xc:vcalendar>
944         <xc:properties>
945             <xc:calscale><text>GREGORIAN</text></xc:calscale>
946             <xc:prodid>
947                 <xc:text>-//Example Inc.//Example Calendar//EN</xc:text>
948             </xc:prodid>
949             <xc:version><xc:text>2.0</xc:text></xc:version>
950         </xc:properties>
951         <xc:components>
952             <xc:vfreebusy>
953                 <xc:properties>
954                     <xc:uid>
955                         <xc:text>76ef34-54a3d2@example.com</xc:text>
956                     </xc:uid>
957                     <xc:dtstart>
958                         <xc:date-time>20070901T000000Z</xc:date-time>
959                     </xc:dtstart>
960                     <xc:dtend>
961                         <xc:date-time>20070931T000000Z</xc:date-time>
962                     </xc:dtend>
963                     <xc:dtstamp>
964                         <xc:date-time>20050530T123421Z</xc:date-time>
965                     </xc:dtstamp>
966                     <xc:freebusy>
967                         <xc:period>20070915T230000Z/20070916T010000Z</xc:period>
968                     </xc:freebusy>
969                 </xc:vfreebusy>
970             </xc:components>
971         </xc:vcalendar>
972     </xc:icalendar>

```

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

```

974 >> Request <<

```

```
975 GET /freebusy/user1@example.com?start=2012-12-01T00:00:00-08:00&end=2012-12-
976 31T00:00:00-08:00
977 HTTP/1.1
978 Host: www.example.com
979
980 >> Response <<
981 HTTP/1.1 404 No data
982
```

983 **11 Conformance**

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

986 Appendix A. Acknowledgments

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

989 **Participants:**

990 Bruce Bartell, Southern California Edison
991 Brad Benson, Trane
992 Edward Cazalet, Individual
993 Toby Considine, University of North Carolina at Chapel Hill
994 William Cox, Individual
995 Sharon Dinges, Trane
996 Mike, Douglass, Rensselaer Polytechnic Institute
997 Craig Gemmill, Tridium, Inc.
998 Girish Ghatikar, Lawrence Berkeley National Laboratory
999 Gerald Gray, Southern California Edison
1000 David Hardin, ENERNOC
1001 Gale Horst, Electric Power Research Institute (EPRI)
1002 Gershon Janssen, Individual
1003 Ed Koch, Akuacom Inc.
1004 Benoit Lepeuple, LonMark International*
1005 Carl Mattocks, CheckMi*
1006 Robert Old, Siemens AG
1007 Alexander Papaspyrou, Technische Universitat Dortmund
1008 Joshua Phillips, ISO/RTO Council (IRC)
1009 Jeremy J. Roberts, LonMark International
1010 David Thewlis, CalConnect
1011

1012 The Calendaring and Scheduling Consortium (CalConnect) TC-XML committee worked closely with WS-
1013 Calendar Technical Committee, bridging to developing IETF standards and contributing the services
1014 definitions that make up Services in Section 4. The Technical Committee gratefully acknowledges their
1015 assistance and cooperation as well. Contributors to TC XML include:

1016 Cyrus Daboo, Apple
1017 Mike Douglass, Rensselaer Polytechnic Institute
1018 Steven Lees, Microsoft
1019 Tong Li, IBM
1020

1021

Appendix B. An Introduction to Internet Calendaring

1022 *The WS-Calendar Technical Committee thanks CalConnect for contributing this overview of iCalendar*
1023 *and its use.*

1024 B.1 icalendar

1025 B.1.1 History

1026 The iCalendar specification was first produced by the IETF in 1998 as RFC 2445 [1]. Since then it has
1027 become the dominant standard for calendar data interchange on the internet and between devices
1028 (desktop computers, mobile phones etc.). The specification was revised in 2009 as RFC 5545 [4].

1029 Alongside iCalendar is the iTIP specification (RFC 2446 [2] and revised as RFC 5546[5]) that defines how
1030 iCalendar is used to carry out scheduling operations (for example, how an organizer can invite attendees
1031 to a meeting and receive their replies). This forms the basis for email-based scheduling using iMIP (the
1032 specification that describes how to use iTIP with email - RFC 6047 [3]).

1033 iCalendar itself is a text-based data format. However, an XML format is also available, providing a one-to-
1034 one mapping to the text format (draft [7]).

1035 iCalendar data files typically have a .ics file name extension. Most desktop calendar clients can import or
1036 export iCalendar data, or directly access such data over the Internet using a variety of protocols.

1037 B.1.2 Data model

1038 The iCalendar data format has a well defined data model. "iCalendar objects" encompass a set of
1039 "iCalendar Components" each of which contains a set of "iCalendar properties" and possibly other sub-
1040 Components. An iCalendar property consists of a name, a set of optional parameters (specified as "key-
1041 value" pairs) and a value.

1042 iCalendar Components include:

1043 "VEVENT" which represents an event

1044 "VTODO" which represents a task or to-do

1045 "VJOURNAL" which represents a journal entry

1046 "VFREEBUSY" which represents periods of free or busy time information

1047 "VTIMEZONE" which represents a timezone definition (timezone offset and daylight saving rules)

1048 "VALARM" is currently the only defined sub-Component and is used to set alarms or reminders on events
1049 or tasks.

1050 Properties include:

1051 "DTSTART" which represents a start time for a Component

1052 "DTEND" which represents an end time for a Component

1053 "SUMMARY" which represents a title or summary for a Component

1054 "RRULE" which can specify rules for repeating events or tasks (for example, every day, every week on
1055 Tuesdays, etc.)

1056 "ORGANIZER" which represents the calendar user who is organizing an event or assigning a task

1057 "ATTENDEE" which represents calendar users attending an event or assigned a task

1058 In addition to this data model and the pre-defined properties, the specification defines how all those are
1059 used together to define the semantics of calendar objects and scheduling. The semantics are basically a
1060 set of rules stating how all the Components and properties are used together to ensure that all iCalendar
1061 products can work together to achieve good interoperability. For example, a rule requires that all events
1062 must have one and only one "DTSTART" property. The most important part of the iCalendar specification

1063 is the semantics of the calendaring model that it represents. The use of text or XML to encode those is
1064 secondary.

1065 **B.1.3 Scheduling**

1066 The iTIP specification defines how iCalendar objects are exchanged in order to accomplish the key task
1067 needed to schedule events or tasks. An example of a simple workflow is as follows:

- 1068 1. To schedule an event, an organizer creates the iCalendar object representing the event and adds
1069 calendar users as attendees.
- 1070 2. The organizer then sends an iTIP "REQUEST" message to all the attendees.
- 1071 3. Upon receipt of the scheduling message, each attendee can decide whether they want to attend
1072 the meeting or not.
- 1073 4. Each attendee can then respond back to the organizer using an iTIP "REPLY" message
1074 indicating their own attendance status.

1075 iTIP supports other types of scheduling messages, for example, to cancel meetings, add new instances to
1076 a repeating meeting, etc.

1077 **B.1.4 Extensibility**

1078 iCalendar was designed to be extensible, allowing for new Components, properties and parameters to be
1079 defined as needed. A registry exists to maintain the list of standard extensions with references to their
1080 definitions to ensure anyone can use them and work well with others.

1081 **B.2 Calendar data access and exchange protocols**

1082 **B.2.1 Internet Calendar Subscriptions**

1083 An Internet calendar subscription is simply an iCalendar data file made available on a web server. Users
1084 can use this data in two ways:

- 1085 – The data can be downloaded from the web server and then imported directly into an iCalendar
1086 aware client. This solution works well for calendar data that is not likely to change over time (for
1087 example the list of national holidays for the next year).
- 1088 – Calendar clients that support "direct" subscriptions can use the URL to the calendar data on the
1089 web server to download the calendar data themselves. Additionally, the clients can check the web
1090 server on a regular basis for updates to the calendar data, and then update their own cached
1091 copy of it. This allows calendar data that changes over time to be kept synchronized.

1092 **B.2.2 CalDAV**

1093 CalDAV is a calendar access protocol and is defined in RFC 4791 [6]. The protocol is based on WebDAV
1094 which is an extension to HTTP that provides enhanced capabilities for document management on web
1095 servers.

1096 CalDAV is used in a variety of different environments, ranging from very large internet service providers,
1097 to large and small corporations or institutions, and to small businesses and individuals.

1098 CalDAV clients include desktop applications, mobile devices and browser-based solutions. It can also be
1099 used by "applets", for example, a web page panel that displays a user's upcoming events.

1100 One of the key aspects of CalDAV is its data model. Simply put, it defines a "calendar home" for each
1101 calendar user, within which any number of "calendars" can be created. Each "calendar" can contain any
1102 number of iCalendar objects representing individual events, tasks or journal entries. This data model
1103 ensures that clients and servers can interoperate well.

1104 In addition to providing simple operations to read, write and delete calendar data, CalDAV provides a
1105 querying mechanism to allow clients to fetch calendar data matching specific criteria. This is commonly

1106 used by clients to do "time-range" queries, i.e., find the set of events that occur within a given start/end
1107 time period.

1108 CalDAV also supports access control allowing for features such as delegated calendars and calendar
1109 sharing.

1110 CalDAV also specifies how scheduling operations can be done using the protocol. Whilst it uses the
1111 semantics of the iTIP protocol, it simplifies the process by allowing simple calendar data write operations
1112 to trigger the sending of scheduling messages, and it has the server automatically process the receipt of
1113 scheduling messages. Scheduling can be done with other users on the CalDAV server or with calendar
1114 users on other systems (via some form of "gateway").

1115 **B.2.3 ActiveSync/SyncML**

1116 ActiveSync and SyncML are technologies that allow multiple devices to synchronize data with a server,
1117 with calendar data being one of the classes of data supported. These have typically been used for low-
1118 end and high-end mobile devices.

1119 **B.2.4 CalWS**

1120 CalWS refers to a set of web services calendar access APIs developed under a cooperative agreement
1121 between The Calendaring and Scheduling Consortium (CalConnect) and OASIS, and being published as
1122 a work product of the WS-Calendar Technical Committee. CalWS defines an API to access and
1123 manipulate calendar data stored on a server. It follows a similar data model to CalDAV and has been
1124 designed to co-exist with a CalDAV service offering the same data.

1125 This specification is part of the CalWS set.

1126 **B.2.5 iSchedule**

1127 iSchedule is a protocol to allow scheduling between users on different calendaring systems and across
1128 different internet domains. It transports iTIP scheduling messages using HTTP between servers. Servers
1129 use DNS and various security mechanisms to determine the authenticity of messages received.

1130 It has been specifically designed to be independent of any calendar system in use at the endpoints, so
1131 that it is compatible with many different systems. This allows organizations with different calendar
1132 systems to exchange scheduling messages with each other, and also allows a single organization with
1133 multiple calendar systems (for example due to mergers, or different departmental requirements) to
1134 exchange scheduling messages between users of each system.

1135 **B.3 References**

1136 [1] <https://datatracker.ietf.org/doc/rfc2445/> : 'Internet Calendaring and Scheduling Core Object
1137 Specification'

1138 [2] <https://datatracker.ietf.org/doc/rfc2446/> : 'iCalendar Transport-Independent Interoperability Protocol'

1139 [3] <https://datatracker.ietf.org/doc/rfc6047/> : 'iCalendar Message-Based Interoperability Protocol'

1140 [4] <https://datatracker.ietf.org/doc/rfc5545/> : 'Internet Calendaring and Scheduling Core Object
1141 Specification'

1142 [5] <https://datatracker.ietf.org/doc/rfc5546/> : 'iCalendar Transport-Independent Interoperability Protocol'

1143 [6] <https://datatracker.ietf.org/doc/rfc4791/> : 'Calendaring Extensions to WebDAV'

1144 [7] <https://datatracker.ietf.org/doc/draft-daboo-et-al-icalendar-in-xml/> : 'xCal: The XML format for
1145 iCalendar'

1146

1147

1148

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. Consistent 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

1149

1150