



WS-Calendar Version 1.0

Committee Draft 01

15 September 2010

Specification URIs:

This Version:

- <http://docs.oasis-open.org/WS-Calendar/v1.0/cd01/WS-Calendar-1.0-spec-cd-01.pdf>
- <http://docs.oasis-open.org/WS-Calendar/v1.0/cd01/WS-Calendar-1.0-spec-cd-01.html>
- <http://docs.oasis-open.org/WS-Calendar/v1.0/cd01/WS-Calendar-1.0-spec-cd-01.doc>

Previous Version:

N/A

Latest Version:

- <http://docs.oasis-open.org/WS-Calendar/v1.0/WS-Calendar-1.0-spec.pdf>
- <http://docs.oasis-open.org/WS-Calendar/v1.0/WS-Calendar-1.0-spec.html>
- <http://docs.oasis-open.org/WS-Calendar/v1.0/WS-Calendar-1.0-spec.doc>

Technical Committee:

OASIS WS-Calendar TC

Chair(s):

Toby Considine

Editor(s):

Toby Considine

Related work:

This specification replaces or supersedes:

N/A

This specification is related to:

- IETF RFC5545, ICalendar
- IETF RFC5546, ICalendar Transport
- IETF RFC2447, ICalendar Message Based Interoperability
- IETF / CalConnect [XCAL] specification in progress
- IETF / CalConnect Calendar Resource Schema specification in progress
-

Declared XML Namespace(s):

<http://docs.oasis-open.org/ws-calendar/>

<http://docs.oasis-open.org/ns/ws-calendar/WS-Calendar-201001>

Abstract:

WS-Calendar describes a limited set of message components and interactions providing a common basis for specifying schedules and intervals to coordinate activities between services. The specification includes service definitions consistent with the OASIS SOA Reference Model and XML vocabularies for the interoperable and standard exchange of:

- Schedules, including sequences of schedules
- Intervals, including sequences of intervals

47 These message components describe schedules and intervals future, present, or past (historical). The
48 definition of the services performed to meet a schedule or interval depends on the market context in
49 which that service exists. It is not in scope for this TC to define those markets or services.

50 Status:

51 This document was last revised or approved by the WS-Calendar Technical Committee on the above
52 date. The level of approval is also listed above. Check the "Latest Version" or "Latest Approved Version"
53 location noted above for possible later revisions of this document.

54 Technical Committee members should send comments on this specification to the Technical Committee's
55 email list. Others should send comments to the Technical Committee by using the "Send A Comment"
56 button on the Technical Committee's web page at <http://www.oasis-open.org/committees/WS-Calendar/>.

57 For information on whether any patents have been disclosed that may be essential to implementing this
58 specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights
59 section of the Technical Committee web page ([http://www.oasis-open.org/committees/WS-](http://www.oasis-open.org/committees/WS-Calendar/ipr.php)
60 [Calendar/ipr.php](http://www.oasis-open.org/committees/WS-Calendar/ipr.php)).

61 The non-normative errata page for this specification is located at [http://www.oasis-](http://www.oasis-open.org/committees/WS-Calendar/)
62 [open.org/committees/WS-Calendar/](http://www.oasis-open.org/committees/WS-Calendar/).

63 **Notices**

64 Copyright © OASIS® 2010. All Rights Reserved.

65 All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual
66 Property Rights Policy (the "OASIS IPR Policy"). The full Policy may be found at the OASIS website.

67 This document and translations of it may be copied and furnished to others, and derivative works that
68 comment on or otherwise explain it or assist in its implementation may be prepared, copied, published,
69 and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice
70 and this section are included on all such copies and derivative works. However, this document itself may
71 not be modified in any way, including by removing the copyright notice or references to OASIS, except as
72 needed for the purpose of developing any document or deliverable produced by an OASIS Technical
73 Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must
74 be followed) or as required to translate it into languages other than English.

75 The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors
76 or assigns.

77 This document and the information contained herein is provided on an "AS IS" basis and OASIS
78 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY
79 WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY
80 OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A
81 PARTICULAR PURPOSE.

82 OASIS requests that any OASIS Party or any other party that believes it has patent claims that would
83 necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard,
84 to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to
85 such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that
86 produced this specification.

87 OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of
88 any patent claims that would necessarily be infringed by implementations of this specification by a patent
89 holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR
90 Mode of the OASIS Technical Committee that produced this specification. OASIS may include such
91 claims on its website, but disclaims any obligation to do so.

92 OASIS takes no position regarding the validity or scope of any intellectual property or other rights that
93 might be claimed to pertain to the implementation or use of the technology described in this document or
94 the extent to which any license under such rights might or might not be available; neither does it
95 represent that it has made any effort to identify any such rights. Information on OASIS' procedures with
96 respect to rights in any document or deliverable produced by an OASIS Technical Committee can be
97 found on the OASIS website. Copies of claims of rights made available for publication and any
98 assurances of licenses to be made available, or the result of an attempt made to obtain a general license
99 or permission for the use of such proprietary rights by implementers or users of this OASIS Committee
100 Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no
101 representation that any information or list of intellectual property rights will at any time be complete, or
102 that any claims in such list are, in fact, Essential Claims.

103 The names "OASIS", [insert specific trademarked names and abbreviations here] are trademarks of
104 OASIS, the owner and developer of this specification, and should be used only to refer to the organization
105 and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications,
106 while reserving the right to enforce its marks against misleading uses. Please see [http://www.oasis-](http://www.oasis-open.org/who/trademark.php)
107 [open.org/who/trademark.php](http://www.oasis-open.org/who/trademark.php) for above guidance.

108

109 Table of Contents

110	1	Introduction.....	8
111	1.1	Terminology.....	8
112	1.2	Normative References.....	8
113	1.3	Non-Normative References.....	10
114	1.4	Naming Conventions.....	10
115	1.5	Architectural References.....	10
116	2	Overview of WS-Calendar.....	11
117	2.1	Approach taken by the WS-Calendar Technical Committee.....	11
118	2.2	Scheduling Service Performance.....	11
119	2.2.1	Which Time? UCT vs. Local Time.....	12
120	2.3	Overview of This Document.....	12
121	3	Intervals, Temporal Relations, and Sequences.....	13
122	3.1	Core Semantics derived from [XCAL].....	13
123	3.1.1	Time.....	13
124	3.2	Intervals.....	13
125	3.2.1	Intervals: the Basic Time Segment.....	14
126	3.3	Temporal Relations between Intervals.....	15
127	3.4	Sequences: Combining Intervals.....	17
128	3.4.1	Scheduling a Sequence.....	18
129	3.5	Alarms.....	19
130	3.6	Time Stamps.....	19
131	3.6.1	Time Stamp Realm Discussion.....	21
132	4	Service Characteristics: Attachments & Performance.....	22
133	4.1	Services and Service Characteristics.....	22
134	4.1.1	Attachments.....	22
135	4.1.2	Specifying Timely Performance.....	23
136	4.1.3	Combining Service and Performance.....	25
137	5	Inheritance and Entry Points: Calendar Gluons.....	27
138	5.1.1	Calendar Gluons.....	27
139	5.1.2	Calendar Gluons and Sequences.....	28
140	5.1.3	Inheritance rules for Calendar Gluons.....	30
141	5.1.4	Optimizing the expression of a Partition.....	31
142	5.1.5	Mixed Inheritance of Start Time.....	35
143	5.1.6	Other Scheduling Scenarios.....	37
144	6	WS-Calendar Models.....	41
145	6.1	Abstract model for WS-Calendar Objects.....	41
146	6.2	Implementation Model for WS-Calendar.....	Error! Bookmark not defined.
147	7	Calendar Services.....	43
148	7.1	Overview of the protocol.....	43
149	7.1.1	Calendar Object Resources.....	43
150	7.1.2	Timezone information.....	43
151	7.1.3	Issues not addressed by this specification.....	44
152	7.1.4	CalWS Glossary.....	44

153	7.2 Error conditions.....	45
154	7.2.1 Example: error with CalDAV error condition	45
155	8 Properties and link relations	46
156	8.1 Property and relation-type URIs	46
157	8.2 supported-features property.	46
158	8.3 max-attendees-per-instance	46
159	8.4 max-date-time	46
160	8.5 max-instances.....	46
161	8.6 max-resource-size	46
162	8.7 min-date-time	47
163	8.8 description.....	47
164	8.9 timezone-service relation.....	47
165	8.10 principal-home relation.	47
166	8.11 current-principal-freebusy relation.	47
167	8.12 principal-freebusy relation.....	47
168	8.13 child-collection relation.	47
169	8.14 created link property	48
170	8.15 last-modified property	48
171	8.16 displayname property	48
172	8.17 timezone property	48
173	8.18 owner property	48
174	8.19 collection link property	48
175	8.20 calendar-collection link property	48
176	8.21 CalWS:privilege-set XML element.....	49
177	9 Retrieving Collection and Service Properties.....	50
178	9.1 Request parameters	50
179	9.2 Responses:	50
180	9.3 Example - retrieving server properties:.....	50
181	10 Creating Calendar Object Resources.....	52
182	10.1 Request parameters	52
183	10.2 Responses:	52
184	10.3 Preconditions for Calendar Object Creation	52
185	10.4 Example - successful POST:.....	53
186	10.5 Example - unsuccessful POST:.....	53
187	11 Retrieving resources.....	54
188	11.1 Request parameters	54
189	11.2 Responses:	54
190	11.3 Example - successful fetch:	54
191	11.4 Example - unsuccessful fetch:.....	54
192	12 Updating resources	55
193	12.1 Responses:	55
194	13 Deletion of resources.....	57
195	13.1 Delete for Collections.....	57
196	13.2 Responses:	57
197	14 Querying calendar resources	58

198	14.1 Limiting data returned	58
199	14.2 Pre/postconditions for calendar queries	58
200	14.3 Example: time range limited retrieval	58
201	15 Free-busy queries.....	63
202	15.1 ACCEPT header	63
203	15.2 URL Query Parameters	63
204	15.2.1 start.....	63
205	15.2.2 end.....	64
206	15.2.3 period.....	64
207	15.2.4 account.....	64
208	15.3 URL parameters - notes	64
209	15.4 HTTP Operations	64
210	15.5 Response Codes	64
211	15.6 Examples	65
212	16 Conformance	68
213	A. Acknowledgements	69
214	B. An Introduction to Internet Calendaring.....	70
215	B.1 icalendar	70
216	B.1.1 History	70
217	B.1.2 Data model.....	70
218	B.1.3 Scheduling	71
219	B.1.4 Extensibility	71
220	B.2 Calendar data access and exchange protocols	71
221	B.2.1 Internet Calendar Subscriptions.....	71
222	B.2.2 CalDAV	71
223	B.2.3 ActiveSync/SyncML	72
224	B.2.4 CalWS.....	72
225	B.2.5 iSchedule	72
226	B.3 References	72
227	C. Overview of WS-Calendar, its Antecedents and its Use	73
228	C.1 Scheduling Sequences	74
229	C.1.1 Academic Scheduling example.....	74
230	C.1.2 Market Performance schedule.....	75
231	Revision History	76
232		

233 **Tables**

234 **Index of Tables**

235 Table 3-1: Defining Time Segments for WS-Calendar 14
236 Table 3-2: VTODO elements in Intervals 14
237 Table 3-3: Temporal Relationships in WS-Calendar 16
238 Table 3-4: Elements of a Temporal Relationship 16
239 Table 3-5: Introducing the Sequence 17
240 Table 3-6: Aspects of Time Stamps 19
241 Table 4-1: Elements of a WS-Calendar Attachment 22
242 Table 4-2: Performance Characteristics 23
243 Table 5-1: Calendar Gluon elements in WS-Calendar 27
244 Table 5-2 Gluon Inheritance rules 30
245
246

247 **Index of Examples**

248 Example 1: An Interval 15
249 Example 3: Temporal Relationship with and without Gap 17
250 Example 4: A Scheduled Sequence 18
251 Example 6: Use of an Attachment with external reference 23
252 Example 8: Interval with inline XML artifact and optional specified Performance 25
253 Example 9: Interval with external reference and optional specified performance 25
254 Example 11: Partition with Duration and Performance defined in the Calendar Gluon 31
255 Example 12: Partition without annotations 33
256 Example 14: Partition with Duration and Performance defined in the Calendar Gluon 35
257 Example 16: Successful Update 55
258 Example 17: Unsuccessful Update 55
259
260

261 1 Introduction

262 One of the most fundamental components of negotiating services is agreeing when something should
263 occur, and in auditing when they did occur. Short running services traditionally have been handled as if
264 they were instantaneous, and have handled scheduling through just-in-time requests. Longer running
265 processes, including physical processes, may require significant lead times. When multiple long-running
266 services participate in the same business process, it may be more important to negotiate a common
267 completion time than a common start time. Pre-existing approaches that rely on direct control of such
268 services by a central system increases integration costs and reduce interoperability as they require the
269 controlling agent to know and manage multiple lead times.

270 Not all services are requested one time as needed. Processes may have multiple and periodic
271 occurrences. An agent may need to request identical processes on multiple schedules. An agent may
272 request services to coincide with or to avoid human interactions. Service performance be required on the
273 first Tuesday of every month, or in weeks in which there is no payroll, to coordinate with existing business
274 processes. Service performance requirements may vary by local time zone. A common schedule
275 communication must support diverse requirements.

276 Physical processes are already being coordinated by web services. Building systems and industrial
277 processes are operated using oBIX, BACnet/WS, LON-WS, OPC XML, and a number of proprietary
278 specifications including TAC-WS, Gridlogix EnNet, and MODBUS.NET. In particular, if building systems
279 coordinate with the schedules of the building's occupants, they can reduce energy use while improving
280 performance.

281 An increasing number of specifications envision synchronization of processes through mechanisms
282 including broadcast scheduling. Efforts to build an intelligent power grid (or smart grid) rely on
283 coordinating processes in homes, offices, and industry with projected and actual power availability;
284 mechanisms proposed include communicating different prices at different times. Several active OASIS
285 Technical Committees require a common means to specify schedule and interval: Energy Interoperation
286 (EITC) and Energy Market Information Exchange (EMIX). Emergency management coordinators wish to
287 inform geographic regions of future events, such as a projected tornado touchdown, using EDXL. The
288 open Building Information Exchange specification [oBIX] lacks a common schedule communications for
289 interaction with enterprise activities. These and other efforts would benefit from a common cross-domain,
290 cross specification standard for communicating schedule and interval.

291 For human interactions and human scheduling, the well-known iCalendar format is used to address these
292 problems. Prior to WS-Calendar, there has been no comparable standard for web services. As an
293 increasing number of physical processes become managed by web services, the lack of a similar
294 standard for scheduling and coordination of services becomes critical.

295 The intent of the WS-Calendar technical committee was to adapt the existing specifications for
296 calendaring and apply them to develop a standard for how schedule and event information is passed
297 between and within services. The standard adopts the semantics and vocabulary of iCalendar for
298 application to the completion of web service contracts. WS Calendar builds on work done and ongoing in
299 The Calendaring and Scheduling Consortium (CalConnect), which works to increase interoperation
300 between calendaring systems.

301 Everything with the exception of all examples, all appendices, and the introduction is normative.

302 1.1 Terminology

303 The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD
304 NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described
305 in

306 Normative References

307 **Calendar Resource Schema** C. Joy, C. Daboo, M Douglas, *Schema for representing*
308 *resources for calendaring and scheduling services,*

309 <http://tools.ietf.org/html/draft-cal-resource-schema-00>, (Internet-Draft),
310 April 2010.

311 **FreeBusy Read URL** E York. *Freebusy read URL*,
312 [http://www.calconnect.org/pubdocs/CD0903%20Freebusy%20Read%20](http://www.calconnect.org/pubdocs/CD0903%20Freebusy%20Read%20URL%20V1.0.pdf)
313 [URL%20V1.0.pdf](http://www.calconnect.org/pubdocs/CD0903%20Freebusy%20Read%20URL%20V1.0.pdf)

314 **RFC2119** S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*,
315 <http://www.ietf.org/RFC/RFC2119.txt>, IETF RFC2119, March 1997.

316 **RFC2447** F. Dawson, S. Mansour, S. Silverberg, *iCalendar Message-Based*
317 *Interoperability Protocol (iMIP)*, <http://www.ietf.org/RFC/RFC2247.txt>,
318 IETF RFC2447, December 2009.

319 **RFC2616** R Fielding, et al. et al, *Hypertext Transfer Protocol -- HTTP/1.1*
320 <http://tools.ietf.org/html/RFC2616>, IETF RFC2616, June 1999

321 **RFC3339** G Klyne, C Newman, *Date and Time on the Internet: Timestamps*
322 <http://tools.ietf.org/html/rfc3339>

323 **RFC4791** Daboo, et al. *Calendaring Extensions to WebDAV (CalDAV)*.
324 <http://www.ietf.org/RFC/RFC4791.txt>. IETF RFC 2119, March 2007

325 **RFC4918** L. Dusseault, *HTTP Extensions for Web Distributed Authoring and*
326 *Versioning (WebDAV)*
327 <http://tools.ietf.org/html/rfc4918>

328 **RFC5545** B. Desruisseaux *Internet Calendaring and Scheduling Core Object*
329 *Specification (iCalendar)*, <http://www.ietf.org/RFC/RFC5545.txt>, IETF
330 RFC5545, September 2009.

331 **RFC5546** C. Daboo *iCalendar Transport-Independent Interoperability Protocol*
332 *(iTIP)*, <http://www.ietf.org/RFC/RFC5546.txt>, IETF RFC5546, January
333 1999.

334 **SOA-RM** OASIS Standard, *Reference Model for Service Oriented Architecture 1.0*,
335 October 2006.
336 <http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.pdf>

337 **Web-Linking** M. Nottingham, *Web linking*. [http://tools.ietf.org/html/draft-nottingham-](http://tools.ietf.org/html/draft-nottingham-http-link-header)
338 [http-link-header](http://tools.ietf.org/html/draft-nottingham-http-link-header) May 2010

339 **draft xCal** C. Daboo, M Douglas, S Lees *xCal: The XML format for iCalendar*,
340 <http://tools.ietf.org/html/draft-daboo-et-al-icalendar-in-xml-03>, Internet-
341 Draft, April 2010.

342 **XPATH** A Berglund, S Boag, D Chamberlin, MF Fernández, M Kay, J Robie, J
343 Siméon *XML Path Language (XPath) 2.0*, <http://www.w3.org/TR/xpath20/>
344 January 2007.

345 **XLINK** S DeRose, E Maler, D Orchard, N Walsh *XML Linking Language (XLink)*
346 *Version 1.1.*, <http://www.w3.org/TR/xlink11/> May 2010.

347 **XPOINTER** S DeRose, E Maler, R Daniel Jr. *XPointer xpointer Scheme*,
348 <http://www.w3.org/TR/xptr-xpointer/> December 2002.

349 **XML SCHEMA** PV Biron, A Malhotra, *XML Schema Part 2: Datatypes Second Edition*,
350 <http://www.w3.org/TR/xmlschema-2/> October 2004.

351 **XRD** OASIS XRI Committee Draft 01, *Extensible Resource Descriptor (XRD)*
352 *Version 1.0*, <http://docs.oasis-open.org/xri/xrd/v1.0/cd01/xrd-1.0-cd01.pdf>
353 October 2009.

354 1.2 Non-Normative References

- 355 **NIST Framework and Roadmap for Smart Grid Interoperability Standards**, Office of the
356 National Coordinator for Smart Grid Interoperability, Release 1.0, NIST
357 Special Publication 1108,
358 http://www.nist.gov/public_affairs/releases/upload/smartgrid_interoperability_final.pdf January 2010.
359
- 360 **NAESB Smart Grid Requirements** (*dunno what reference I need here*)
361
- 362 **REST** T Fielding, *Architectural Styles and the Design of Network-based*
363 *Software Architectures*,
364 <http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>.
- 365 **TZDB** P Eggert, A.D. Olson, "Sources for Time Zone and Daylight Saving Time
366 Data", July 2009, <http://www.twinsun.com/tz/tz-link.htm>
- 367
- 368 **Time Zone Recommendations**, CalConnect, *CalConnect EDST (Extended Daylight*
369 *Savings Time) Reflections and Recommendations*, Version: 1.1,
370 <http://www.calconnect.org/pubdocs/CD0707%20CalConnect%20EDST%20Reflections%20and%20Recommendations%20V1.1.pdf>
371
372 2007-10-04
- 373 **Time Zone Service**, M Douglas, C Daboo, *Timezone Service Protocol*, Draft RFC,IETF,
374 <http://datatracker.ietf.org/doc/draft-douglass-timezone-service/>
375 2007-07-05
376

377 1.3 Naming Conventions

- 378 This specification follows some naming conventions for artifacts defined by the specification, as follows:
379 For the names of elements and the names of attributes within XSD files, the names follow the
380 camelCase¹ convention, with all names starting with a lower case letter, eg
381

```
<element name="componentType" type="WS-Calendar:ComponentType"/>
```
- 382 For the names of types within XSD files, the names follow the CamelCase convention with all names
383 starting with an upper case letter, e.g.,
384

```
<complexType name="ComponentService">
```
- 385 For the names of intents, the names follow the CamelCase convention, with all names starting with a
386 lower case letter, EXCEPT for cases where the intent is to represent an established acronym, in which
387 case the entire name follows the usage of the established acronym.
- 388 An example of an intent which references an acronym is the "SOAP" intent.

389 1.4 Architectural References

- 390 WS-Calendar assumes incorporation into services. Accordingly it assumes a certain amount of definitions
391 of roles, names, and interaction patterns. This document relies heavily on roles and interactions as
392 defined in the OASIS Standard *Reference Model for Service Oriented Architecture [SOA-RM]*.

¹ Common term - see Wikipedia for explanation.

393 2 Overview of WS-Calendar

394 A calendar communication without a real world effect² is of little interest. That real world effect is the result
395 of a services execution context within a policy context. Practitioners can use WS-Calendar to add
396 communication of schedule and interval to the execution context of a service. Use of WS-Calendar will
397 align the performance expectations between execution contexts in different domains. The Technical
398 Committee intends for other specifications and standards to incorporate WS-Calendar, bringing a
399 common scheduling context to diverse interactions in different domains

400 2.1 Approach taken by the WS-Calendar Technical Committee

401 The Technical Committee (TC) based its work upon the iCalendar specification as updated in 2009 (IETF
402 **[RFC5545]** and its the XML serialization **[XCAL]**, currently (2010-07) on a standards track in the IETF.
403 Members of the Calendaring and Scheduling Consortium (CalConnect.org) developed both updates to
404 IETF specifications and provided advice to this TC. This work provides the vocabulary for use in this
405 specification.

406 The committee solicited requirements from a range of interests, notably the NIST Smart Grid Roadmap
407 and the requirements of the Smart Grid Interoperability Panel (SGIP) as developed by the North
408 American Energy Standards Board (NAESB). Others submitting requirements included members of the
409 oBIX technical committee and representative of the FIX Protocol Association. These requirements are
410 reflected in the semantic elements described in Chapters 3 and 4.

411 In a parallel effort, the CalConnect TC-XML committee developed a number of schedule and calendar-
412 related services. CalConnect drew on its experience in interoperability between enterprise calendaring
413 systems as well as interactions with web-based calendars and personal digital assistants (PDAs). These
414 services were developed as RESTfull (using **[REST]**)services by CalConnect and contributed to the WS-
415 Calendar TC.

416 2.2 Scheduling Service Performance

417 Time semantics are critical to WS-Calendar. Services requested differently can have different effects on
418 performance even though they appear to request the same time interval. This is inherent in the in the
419 concept of a service oriented architecture.

420 As defined in the OASIS Reference Model for Service Oriented Architecture 1.0³, service requests access
421 the capability of a remote system.

422 *The purpose of using a capability is to realize one or more real world effects. At its core, an*
423 *interaction is “an act” as opposed to “an object” and the result of an interaction is an effect (or a*
424 *set/series of effects). This effect may be the return of information or the change in the state of*
425 *entities (known or unknown) that are involved in the interaction.*

426 *We are careful to distinguish between public actions and private actions; private actions are*
427 *inherently unknowable by other parties. On the other hand, public actions result in changes to the*
428 *state that is shared between at least those involved in the current execution context and possibly*
429 *shared by others. Real world effects are, then, couched in terms of changes to this shared state*

430 A request for remote service performance is a request for specific real world effects. Consider two service
431 providers that offer the same service. One must start planning an hour or more in advance. The second
432 may be able to achieve the service in five minutes. The service start time is the time when that service

² This paragraph includes a number of terms of art used in service oriented architecture (SOA). In all cases, the terms are as defined in the *Reference Model for Service Oriented Architecture*, found in the normative references.

³ See normative references in section 0

433 becomes available. If we do not distinguish these circumstances, then the customer would receive quite
434 different quite different services with no distinctions in the service contract.

435 The complement of this is the scheduled end time. The party offering the service may need to ramp down
436 long running processes. Using for example energy demand response, if a system contracts to end energy
437 use by 3:00, it assumes the onus of turning everything off before 3:00.

438 Duration is how long a behavior is continued. If a service contracts to provide shed load for an hour, it is
439 not necessary for it to stop shedding load 65 minutes later (which may be the end of the work day). It
440 must, however, shed the agreed upon load during all of the 60 minutes.

441 In this way, the service scheduled to shed load from 4:00 ending at 5:00 may be quite different than the
442 one scheduled to shed load for an hour beginning at 4:00.

443 **2.2.1 Which Time? UCT vs. Local Time**

444 When 2 or more parties attempt to agree on a time - e.g., for a meeting, or when to provide a service,
445 they agree to start at a particular instant of time UTC. They agree on that instant in time by converting
446 from local time, e.g., they want a meeting to start at 13:00 Eastern, 18:00 UK. Our lives and the use of
447 services are bound by local time not by UTC. To humans local time is the invariant and UTC is mapped
448 on to it. If a government modifies the rules we adjust the mappings and we shift the UTC time. We still
449 want to meet at 13:00 local or have the heating start at 07:00.

450 As long as the rules never change this causes no confusion—but they do. Recent experience has
451 included considerable efforts when the rules for the start of Daylight Savings Time (DST) have changed.
452 If all information is in UTC, and no record of the events basis in the local time and time zone remains,
453 there is no way to re-compute existing contracts. We don't know if that UTC was calculated based on an
454 old or new rule.

455 A triplet of Local time + timezoneid + (UTC or offset) always allows you to determine if the time is valid. If
456 a recalculation of UTC for that local time + tzid results in a different value from that stored then
457 presumably the DST rules have changed since the data was stored. If you can detect that the scheduled
458 time is no longer valid you can take corrective action.

459 For simplicity, all examples and discussion in this document are based on Greenwich Mean Time also
460 known as Coordinated Universal Time (UTC). The Technical Committee makes no representation as
461 whether UTC or local time are more appropriate for a given interaction. Because WS-Calendar is based
462 on **[iCalendar]**, business practices built upon WS-Calendar can support either.

463 Practitioners should consult **[Time Service Recommendations]** and **[Time Zone Service]** in the non-
464 normative references.

465 **2.3 Overview of This Document**

466 The specification consists of a standard schema and semantics for schedule and interval information.
467 Often the most important service schedule communications involve series of related services over time,
468 which WS-Calendar defines as a Sequence. These semantic elements are defined and discussed in
469 Section 3.

470 Section 4 introduces notions of performance, i.e. what does it mean to be “on time”. This section also
471 describes the different ways to association a service request with each Interval in a Sequence.

472 Managing information exchanges about a Sequence of events can easily become cumbersome, or prone
473 to error. WS-Calendar defines the Calendar Gluon, a mechanism for making assertions about all or most
474 of the intervals in a sequence. Intervals can inherit from a Calendar Gluon, or they can override locally
475 assertions inherited from the Calendar Gluon. Section 5 discusses inheritance and parsimony of
476 communication and introduces contract scheduling.

477 In Sections 7-15, this document describes **[REST]**-based, (RESTfull) web services for interacting with
478 remote calendars. These interactions are based upon the larger iCalendar message. The specification
479 defines services for calendar inquiries, event scheduling, event updating, and event cancelation.

480 3 Intervals, Temporal Relations, and Sequences

481 WS-Calendar Elements are semantic elements derived from the [XCAL] specification. These elements
482 are smaller than a full schedule interaction, and describe the intervals, durations, and time-related events
483 that are relevant to service interactions. The Elements are used to build a precise vocabulary of time,
484 duration, sequence, and schedule.

485 WS-Calendar elements elaborate the objects defined in iCalendar, to make interaction requirements
486 explicit. For example, in human schedule interactions, different organizations have their own
487 expectations. Meetings may start on the hour or within 5 minutes of the hour. As agents scheduled in
488 those organizations, people learn the expected precision. In WS-Calendar, that precision must be explicit
489 to prevent interoperability problems. WS-Calendar defines a performance element to elaborate the simple
490 specification of [XCAL] to make explicit the performance expectations within a scheduled event.

491 WS-Calendar defines common semantics for recording and exchanging event information.

492 3.1 Core Semantics derived from [XCAL]

493 The iCalendar data format [RFC5545] is a widely deployed interchange format for calendaring and
494 scheduling data. The [XCAL] specification (in process) standardizes the XML representation of iCalendar
495 information. WS-Calendar relies on [XCAL] standards and data representation to develop its semantic
496 components.

497 <http://ietfreport.isoc.org/idref/draft-daboo-et-al-icalendar-in-xml/>

498 3.1.1 Time

499 Time is an ISO 8601 compliant time string with the optional accompaniment of a duration interval to
500 define times of less than 1 second. Examples of the from the ISO 8601 standard include:

```
501 Year:  
502     YYYY (eg 1997)  
503 Year and month:  
504     YYYY-MM (eg 1997-07)  
505 Complete date:  
506     YYYY-MM-DD (eg 1997-07-16)  
507 Complete date plus hours and minutes:  
508     YYYY-MM-DDThh:mmTZD (eg 1997-07-16T19:20+01:00)  
509 Complete date plus hours, minutes and seconds:  
510     YYYY-MM-DDThh:mm:ssTZD (eg 1997-07-16T19:20:30+01:00)  
511 Complete date plus hours, minutes, seconds and a decimal fraction of a second  
512     YYYY-MM-DDThh:mm:ss.sTZD (eg 1997-07-16T19:20:30.45+01:00)
```

513 Normative information on [ISO 8601] is found in section 0.

514 The iCalendar Components (VComponents)

515 iCalendar and [XCAL] have a number of long defined component objects that comprise the payload
516 inside of an iCalendar message. These include the VTOD, the VALARM, the VEVENT. These element
517 names begin with "V" for historic reasons. The definitions and use of each of the vObjects is described in
518 [RFC5545].

519 Because of its flexibility, the VTOD object is the basis for WS-Calendar objects for service performance.
520 Because WS-Calendar services support all traditional iCalendar-based interactions (CalDAV, et al.), all
521 VComponents SHALL be supported.

522 3.2 Intervals

523 Time Segments, i.e., increments of continuous passage of time, are a critical component of service
524 alignment using WS-Calendar. There are many overloaded uses of terms about time, and within a

525 particular time segment, there may be many of them. Within this document, we use the term Time
 526 Segments to encompass all the terms in Table 3-1, below.

527 The base data type for time segments is the Interval. The Interval is a time segment defined by the
 528 Duration element as defined in [XCAL]. The [XCAL] duration is a data type based upon the string
 529 representation in the iCalendar duration. The Committee listened to arguments that we should redefine
 530 the use and meaning of Duration. Whatever their merit, the iCalendar Duration has a pre-existing
 531 meaning of the length of time of scheduled within an event. In this section, the Duration is enumerated as
 532 one of several time segments.

533 *Table 3-1: Defining Time Segments for WS-Calendar*

Time Segment	Definition
Duration	Well-known element from iCalendar and [XCAL], Duration is the length of an event scheduled using iCalendar or any of its derivatives. The [XCAL] duration is a data type using the string representation defined in the iCalendar duration. The Duration is the sole descriptive element of the VTODO object that is mandatory in the Interval.
Interval	The Interval is a single duration supported by the full information set of the VTODO object as defined in iCalendar ([RFC5545]) and refined in [XCAL]. A WS-Calendar interval must include a Duration.
Sequence	A Sequence is a set of Intervals with defined temporal relationships. Sequences may have gaps between Intervals, or even simultaneous activities. A sequence is re-locatable, i.e., it does not have a specific date and time. A Sequence may consist of a single interval.
Scheduled Sequence	A Scheduled Sequence is a Sequence that is anchored by a specific date and time, that is, it is a Sequence with a start date and time. Specific performance of a Sequence against a service contract always occurs in a Scheduled Sequence.
Partition	A Partition is a set of consecutive intervals. A Partition includes the trivial case of a single Interval. A Partition is used to define a single service or behavior which varies over time. Examples include energy prices over time and or energy usage over time. A Partition is re-locatable, i.e., it does not have a specific date and time.
Scheduled Partition	A Scheduled Partition is a Partition that is anchored by a specific date and time, that is, it is a Partition with a start date and time. The Performance of a Partition against an executed service contract always occurs in a Scheduled Partition.

534 1.1.1 Intervals: the Basic Time Segment

535 An interval specifies how long an activity lasts. An Unscheduled Interval is not linked to a specific date
 536 and time. Intervals are derived from the [iCalendar] Component VTODO. For ease of reference, the
 537 required elements from the VTODO object are summarized here. Nothing in this section supersedes
 538 [RFC5545] or the [XCAL] specification. Implementers SHALL refer to those respective specifications
 539 [RFC5545] and the [XCAL] specifications for the normative description and definitions.

540 While all elements of the VTODO component are legal in WS-Calendar, certain elements are critical when
 541 invoking services. These elements and their definitions within WS-Calendar are listed in *Table 3-2:*
 542 *VTODO properties in Intervals.*

543 *Table 3-2: VTODO properties in Intervals*

Elements	Use	Use in WS-Calendar
dtstamp	Mandatory	

Elements	Use	Use in WS-Calendar
x-wscalendarType	Mandatory xs:string, value always "Interval"	Added vtodo attribute, ignored by iCalendar processors
uid	Mandatory	Used to enable unambiguous referencing by other components
dtstamp	xcal:dtstamp Optional	Identifies when Interval object was created
duration	xcal:duration Optional	Identifies length of time for Interval
dtStart	Optional	Scheduled start date and time for interval
dtEnd	Ignored	Legal for compatibility only. WS-Calendar does not use dtend.
attach	Mandatory, Multipleoccurs	In [xCal], any attachment. In WS-Calendar, restricted to the Attachment object as defined in section 4.
x-wscalendarrelation	temporalRelation. Optional compound element	Defines temporal relations to other components. Temporal Relations and their use to define Sequences are described in section 3.3.

544 An interval specifies how long an activity lasts. An Unscheduled Interval is not linked to a specific date
545 and time. The example below shows the components section of a WS-Calendar event containing a single
546 interval

547 *Example 1: An Interval*

```

548 <components>
549 <vtodo>
550   <properties>
551     <x-wscalendartype>Interval</x-wscalendartype>
552     <uid>
553       <text>00959BC664CA650E933C892C@example.com</text>
554     </uid>
555     <description>
556       <text>Sample Contract</text>
557     </description>
558     <duration>
559       <duration>
560         <duration>T10H</duration>
561       </duration>
562     </duration>
563   </properties>
564 </vtodo>
565 </components>

```

566 Note that no start time is specified, and no relationship. Relationships are not mandatory until an interval
567 is incorporated into a Sequence.

568 3.3 Temporal Relations between Intervals

569 Many iCalendar communications involve more than one vComponent. In iCalendar interactions there are
570 few components they have stereotypical interactions. For example, a vAlarm may be associated with a
571 vevent. The registered relationships for iCalendar components are PARENT and Child. In [XCAL], these
572 are usually expressed as:

```

573 <relationship>
574   <uid>aaaaaaaa1</uid>
575   <reltype>PARENT</reltype>

```

576

```
</relationship>
```

577 WS-Calendar defines additional relationships to describe how intervals relate in time. These Temporal
578 Relationships express the order of performance and to declare the spacing between any two intervals.
579 These relationships are referred to as the temporal relationships between components.

580

Table 3-3: Temporal Relationships in WS-Calendar

Temporal Relationship	Short Form	Definition
finishToStart	FS	As soon as the related Component finishes, this interval begins.
finishToFinish	FF	Used without gap when two components must finish at the same time. If there is a gap, it indicates that the referring component will finish execution a duration after the referred-to component.
startToFinish	SF	This component must Finish before the related component starts.
startToStart	SS	These Components must start at the same time
Gap		Attribute to indicate the separation, if any, between the state of the first Interval and the state of the second. Expressed as a duration.

581 WS-Calendar specifies more elements in the Relationship to accommodate the needs of Temporal
582 Relationships. WS-Calendar also extends iCalendar relationship to allow references to external
583 Components as well as to those internal to the iCalendar object.

584

Table 3-4: Elements of a WS-Calendar Temporal Relationship

Relationship Element		Definition
Type	String, Mandatory	Enumerated list from union of iCalendar and WS-Calendar Temporal Relationships.
Reference	xcal:uid or xpointer	Identifier of Component in Components collection (if uid) or to external interval (if xpointer).
Gap	xcal:duration <i>Optional</i>	Attribute to indicate the separation, if any, between the state of the first Interval and the state of the second. Expressed as a duration. Only used with Temporal Relationships

585 The relationship below indicates that this Interval is to start ten minutes following the finish of the interval
586 specified.

587

Example 2: Temporal Relationship

588
589
590
591
592
593
594
595
596
597
598

```
<x-wscalendarrelation>
<temporalrelationshipstype>finishtostart</temporalrelationshipstype>
<gap>
  <duration>
    <xcal:duration>T00:10</xcal:duration>
  </duration>
</gap>
<relatedto>
  <uid>00959BC664CA650E933C892C@example.com</uid>
</relatedto>
</x-wscalendarrelation>
```

599 If there is no temporal separation between Intervals, the gap element is optional. The following examples
600 are equivalent expressions to express a relationship wherein both intervals must start at the same
601 moment.

602

Example 3: Temporal Relationship with and without Gap

603
604
605
606
607
608
609
610
611
612
613

```
<x-wsalendarrelation>
<temporalrelationshiptype>starttostart</temporalrelationshiptype>
<gap>
  <duration>
    <xcal:duration>T00:10</xcal:duration>
  </duration>
</gap>
<relatedto>
  <uid>00959BC664CA650E933C892C@example.com</uid>
</relatedto>
</x-wsalendarrelation>
```

614 Leaving out the optional Gap element, we have:

615
616
617
618
619
620

```
<x-wsalendarrelation>
<temporalrelationshiptype>starttostart</temporalrelationshiptype>
<relatedto>
  <uid>00959BC664CA650E933C892C@example.com</uid>
</relatedto>
</x-wsalendarrelation>
```

621 The two expressions of a Temporal Relationship above are equivalent.

622 Intervals with Temporal Relationships enable the message to express complex temporal relations within a
623 Sequence, as well as express the simple consecutive intervals named a Partition

624 As the rules for parsing XML do not mandate preservation of order within a sub-set, we cannot assume
625 that order is preserved when parsing a set of Components. For Sequences, mere order is not enough—
626 each Interval must either refer to or be referred by at least one interval. Either way, a Sequence defines a
627 coherent set of intervals that can be assembled out of members of a collection of intervals

628 3.4 Sequences: Combining Intervals

629 Section 3.3 introduced Temporal Relationships. A collection of intervals with a coherent set of Temporal
630 Relationships is a Sequence. Temporal Relationships are transitive, so that if Interval A is related to
631 Interval B, and Interval B is related to Interval C, then Interval A is related to Interval C.

632 Table 3-5: Introducing the Sequence

633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656

```
<components>
<vtodo>
  <properties>
    <x-wsalendaratype>Interval</x-wsalendaratype>
    <xcal:uid>
      <xcal:text>10959BC664CA650E933C892C@example.com</xcal:text>
    </xcal:uid>
    <xcal:description>
      <xcal:text>First Interval in Sequence</xcal:text>
    </xcal:description>
    <xcal:duration>
      <xcal:duration>T1H</xcal:duration>
    </xcal:duration>
  </properties>
</vtodo>
<vtodo>
  <properties>
    <x-wsalendaratype>Interval</x-wsalendaratype>
    <xcal:uid>
      <xcal:text>20959BC664CA650E933C892C@example.com</xcal:text>
    </xcal:uid>
    <xcal:description>
      <xcal:text>Second Interval in Sequence</xcal:text>
    </xcal:description>
```

```

657     <xcal:summary>
658         <xcal:text>Note the Temporal Relation to the First
659             Interval</xcal:text>
660     </xcal:summary>
661     <xcal:duration>
662         <xcal:duration>T15M</xcal:duration>
663     </xcal:duration>
664     <x-wscalendarrelation>
665         <temporalrelationshiptype>finishtostart</temporalrelationshiptype>
666         <relatedto>
667             <uid>10959BC664CA650E933C892C@example.com</uid>
668         </relatedto>
669     </x-wscalendarrelation>
670 </properties>
671 </vtodo>
672 <vtodo>
673     <properties>
674         <x-wscalendartype>Interval</x-wscalendartype>
675         <xcal:uid>
676             <xcal:text>30959BC664CA650E933C892C@example.com</xcal:text>
677         </xcal:uid>
678         <xcal:description>
679             <xcal:text>Third Interval in Sequence</xcal:text>
680         </xcal:description>
681         <xcal:duration>
682             <xcal:duration>T30M</xcal:duration>
683         </xcal:duration>
684         <x-wscalendarrelation>
685             <temporalrelationshiptype>finishtostart</temporalrelationshiptype>
686             <relatedto>
687                 <uid>20959BC664CA650E933C892C@example.com</uid>
688             </relatedto>
689             <gap>
690                 <xcal:duration>
691                     <xcal:duration>T10M</xcal:duration>
692                 </xcal:duration>
693             </gap>
694         </x-wscalendarrelation>
695     </properties>
696 </vtodo>
697 </components>

```

698 In this example, the Intervals are one hour, 15 minutes, and 30 minutes long. There is a ten minute period
699 between the second and third periods.

700 3.4.1 Scheduling a Sequence

701 A Sequence becomes a Scheduled Sequence whenever single interval within the sequence is scheduled.
702 An interval is scheduled when it has a specific starting time (dtstart).

703 *Example 4: A Scheduled Sequence*

```

704 <components>
705 <vtodo>
706     <properties>
707         <x-wscalendartype>Interval</x-wscalendartype>
708         <xcal:uid>
709             <xcal:text>10959BC664CA650E933C892C@example.com</xcal:text>
710         </xcal:uid>
711         <xcal:description>
712             <xcal:text>First Interval in Sequence</xcal:text>
713         </xcal:description>
714         <dtstart>2010-09-11T13:00</dtstart>
715         <xcal:duration>

```

```

716         <xcal:duration>T1H</xcal:duration>
717     </xcal:duration>
718 </properties>
719 </vtodo>
720 <vtodo>
721     <properties>
722         <x-wscalendartype>Interval</x-wscalendartype>
723         <xcal:uid>
724             <xcal:text>20959BC664CA650E933C892C@example.com</xcal:text>
725         </xcal:uid>
726         <xcal:description>
727             <xcal:text>Second Interval in Sequence</xcal:text>
728         </xcal:description>
729         <xcal:duration>
730             <xcal:duration>T15M</xcal:duration>
731         </xcal:duration>
732         <x-wscalendarrelation>
733             <temporalrelationshipstype>finishtostart</temporalrelationshipstype>
734             <relatedto>
735                 <uid>10959BC664CA650E933C892C@example.com</uid>
736             </relatedto>
737         </x-wscalendarrelation>
738     </properties>
739 </vtodo>
740 </components>

```

741 Note that the entire Sequence is scheduled when a single Interval within the Sequence is scheduled.

742 3.5 Alarms

743 Alarms in WS-Calendar declare when to send notifications between services. Within a single service,
744 alarms declare milestones and target times. The base iCalendar object for all alarms is the VALARM
745 object. This section discusses how the iCalendar VALARM object is used in WS-Calendar.

746 The use of Alarms in enterprise scheduling is a rapidly changing area as this is written, and alarm
747 mechanisms are out of scope for this document. An Alarm notifies another party that something has
748 happened or is about to happen. Some alarms, such as alarm clocks, are scheduled explicitly. Others
749 arise as a notification from another system. WS-Eventing, oBIX alarms, and CAP and EDXL alerts are
750 just a few of the already defined mechanisms.

751 In WS-Calendar, an alarm is a VALARM object within an Interval object, Its actions are [XPOINTER]
752 references to the service or event that is triggered. Valarm also supports recurring activities. A long-
753 running Interval service could include a recurring call-out to a 3rd service providing observation of the
754 service's effects. For example, a Demand Response service could be launched accompanied by a
755 recurring 5 minute request to read the meter from another service.

756 3.6 Time Stamps

757 Time stamps are used everywhere in inter-domain service performance analysis and have particular use
758 in smart grids to support event forensics. Time stamps are often assembled and collated from events
759 across multiple time zones and from multiple systems.

760 Different systems may track time and therefore record events with different levels of Tolerance. It is not
761 unusual for a time stamp from a domain with a low Tolerance to appear to have occurred after events
762 from a domain with high-Tolerance time-stamps that it caused. A fully qualified time-stamp includes the
763 granularity measure.

764 *Table 3-6: Aspects of Time Stamps*

Time Stamp Element	Definition (Normative)	Note (Non-Normative)
--------------------	---------------------------	-------------------------

Time Stamp Element	Definition (Normative)	Note (Non-Normative)
timestamp	WS-Calendar:time A fully qualified date and time of event. Mandatory.	May include two objects as defined above.
precision	A Duration defining the accuracy of the TimeStamp value. Mandatory.	Identifies whether one hour interval is indeed one hour or plus or minus some number of milliseconds, seconds and minutes.
timeStampRealm	Of type Uri, shall identify the system where the TimeStamp value originated. The value of this element shall be set by: <ul style="list-style-type: none"> • The component at the realm border in a particular inter-domain interaction or, • By any component able to accurately set it within a system or sub-system. In the latter case, nothing prevents the component at the realm border to overwrite it without any notice. Optional.	A set of points originating from the same realm are reasonably synchronized. Within a realm, one can assume that time-stamped objects sorted by time are in the order of their occurrence. Between realms, this assumption is rebuttable. A system border is crossed in an interaction when the 2 communication partners are not synchronized based on the same time source. See the example below for more information.
leapSecondsKnown	Xs:bool If True, shall indicate that the TimeStamp value takes into account all leap seconds occurred. Otherwise False. Optional.	Indicates that the time source of the sending device support leap seconds adjustments.
clockFailure	xs:bool If True, shall indicate a failure on the time source preventing the TimeStamp value issuer from setting accurate timestamps. Otherwise False. Mandatory.	Indicates that the time source of the sending device is unreliable. The timestamp should be ignored.
clockNotSynchronized	xs:bool If True, shall indicate the time source of the TimeStamp value issuer is not synchronized correctly, putting in doubt the accuracy of the timestamp. Mandatory.	Indicates that the time source of the sending device is not synchronized with the external UTC time source.

Time Stamp Element	Definition (Normative)	Note (Non-Normative)
timeSourceAccuracy	A Duration defining the accuracy of the time source used in the TimeStampRealm system. Optional.	Represents the time accuracy class of the time source of the sending device relative to the external UTC time source.

765 **3.6.1 Time Stamp Realm Discussion**

766 Within a single system, or synchronized system of systems, one can sort the temporal order of event by
767 sorting them by TimeStamp. Determining the order of events is the first step of event forensics. This
768 assumption does not apply when events are gathered across systems.

769 Different systems may not have synchronized time, or may synchronize time against different sources.
770 This means different system clocks may drift apart. It may be that a later timestamp from one system
771 occurred before an earlier timestamp in another. As this drift is unknown, it cannot be automatically
772 corrected for without additional information.

773 The TimeStampRealm element identifies which system created an event time-stamp. The
774 TimeStampRealm identifies a source system in inter-domain interactions (a system of systems). For
775 example: <http://SystemA.com> and <http://SystemB.com> identify 2 systems. This example assumes
776 SystemA and SystemB do not have a common time source.

777 The TimeStampRealm can also be used to identify sub-systems in intra-domain interactions (sub-systems
778 of a system). For For example: <http://SystemA.com/SubSystem1> and <http://SystemA.com/SubSystem2>
779 identify 2 subsystems of the same higher level system. In case the upper level SystemA does not have a
780 global time source for synchronizing all of its sub-system, it can be useful to identify sub-systems in such
781 a way.

782 **4 Service Characteristics: Attachments &**
 783 **Performance**

784 **4.1 Services and Service Characteristics**

785 While iCalendar expresses time and intervals, WS-Calendar associates those intervals with specific
 786 services and service characteristics. WS-Calendar uses the ATTACH element that is already part of each
 787 iCalendar components to specify services and performance characteristics.

788 In iCalendar, the ATTACH element carries unstructured information associated with the event or alarm
 789 communication. Attachments in iCalendar can also be in the form of URIs pointing outside the iCalendar
 790 structure. WS-Calendar uses structured XML to communicate service intents.

791 **4.1.1 Attachments**

792 The XML artifact in the attachment may be in-line, i.e., contained within the ATTACH element of the
 793 VTODO or VALARM object, or it may be found in another section of the same XML object, sharing the
 794 same message as WS-Calendar element, or it may be discovered by external reference. Attachments,
 795 then, are used to request “perform as described here”, or “perform as described below”, or “perform as
 796 described elsewhere.”

797 The ATTACH element in WS-Calendar has three elements as below.

798 *Table 4-1: Elements of a WS-Calendar Attachment*

Attachment Element	Use	Discussion
artifact	any in-line XML (xs:any). <i>Optional.</i> An attachment must have at least one artifact or reference	Defined per the business process associated with this interaction. WS-Calendar. This is not an object, it is merely a name for use in documentation An attachment must have at least one of
reference	[XPOINTER] <i>Optional</i> An attachment must have at least one of artifact or reference	Points to external XML, or XML located elsewhere in document
performance	WsCalendar:Performance <i>Optional</i>	Specifies time-related performance characteristics.

799 When a WS-Calendar reference uses an external reference to specify a service, that reference is an
 800 object of the type [XPOINTER] (see section 0)..[XPOINTER] is a general purpose URI and XML traversal
 801 standard. This [XPOINTER] object is in the named data element “Reference.”

802 *Example 5: Use of an Attachment with inline XML artifact*

```

803 <vtodo>
804   <properties>
805     <x-wscalendartype>Interval</x-wscalendartype>
806     <xcal:uid>
807       <xcal:text>00959BC664CA650E933C892C@example.com</xcal:text>
808     </xcal:uid>
809     <xcal:description>
810       <xcal:text>Sample Contract</xcal:text></xcal:description>
811     <attach>
812       <artifact>
  
```

```

813         <emix-wip>
814             <price>8.45</price>
815             <quantity>8.45</quantity>
816         </emix-wip>
817     </artifact/>
818 </attach>
819 </properties>
820 </vtodo>

```

821 Note: as this is written, there is no EMIX specification. The Artifact is of type xs:any, allowing compliant
822 XML from any namespace to be used.

823 *Example 6: Use of an Attachment with external reference*

```

824 <vtodo>
825     <properties>
826         <x-wscalendartype>Interval</x-wscalendartype>
827         <xcal:uid>
828             <xcal:text>00959BC664CA650E933C892C@example.com</xcal:text>
829         </xcal:uid>
830         <xcal:description>
831             <xcal:text>Sample Contract</xcal:text></xcal:description>
832         <attach>
833             <reference>http://scheduled.ws-calendar-
834                 service.com/contract1</reference>
835         </attach>
836     </properties>
837 </vtodo>

```

838

839 4.1.2 Specifying Timely Performance

840 Service coordination between systems requires precise communication about expectation for the
841 timeliness of performance. These expectations can be set for each interval or for an entire sequence.
842 This communication is through the performance component of the Attachment.

843 The Performance component refines the meaning of time-related service communication. All elements of
844 the Performance object use the Duration element as defined in [RFC5545].

845 *Table 4-2: Performance Characteristics*

Performance Characteristic	Definition	Discussion
startBeforeTolerance	A Duration enumerating how far before the requested start time the requested service may commence.	Indicates if a service that begins at 1:57 is compliant with a request to start at 2:00
startAfterTolerance	A Duration enumerating how far after the requested start time the requested service may commence.	Indicates if a service that begins at 2:01 is compliant with a request to start at 2:00
endBeforeTolerance	A Duration enumerating how far before scheduled end time may end.	Indicates if a service that ends at 1:57 is compliant with a request to end at 2:00
endAfterTolerance	A Duration enumerating how far after the scheduled end time the requested service may commence.	Indicates if a service that ends at 2:01 is compliant with a request to end at 2:00

Performance Characteristic	Definition	Discussion
durationLongTolerance	A Duration indicating by how much the performance duration may exceed the duration specified in the Interval . It may be 0.	Used when run time is more important than start and stop time. DurationLongTolerance SHALL NOT be used when Start and End Tolerances are both specified.
durationShortTolerance	A Duration indicating by how much the performance duration may fall short of duration specified in the Interval . It may be 0.	Used when run time is more important than start and stop time. DurationShortTolerance SHALL NOT be used when Start and End Tolerances are both specified.
granularity	A Duration enumerating the smallest unit of time measured or tracked	Whatever the time tolerance above, there is some minimum time that is considered insignificant. A Granularity of 1 second defines the tracking and reporting requirements for a service.

846 Performance is part of the core WS-Calendar service definition. Similar products or services, identical
847 except for different Performance characteristics may appear in different markets. Performance
848 characteristics influence the price offered and the service selected.

849 Note that Performance object does not indicate time, but only duration. A performance object associated
850 with an unscheduled Interval does not change when that Interval is scheduled.

851 The Performance object is an optional component of each WS-Calendar attachment.

852 *Example 7: Performance Component*

```

853 <attach>
854   <uri>http://scheduled.ws-calendar-service.com/contract1</uri>
855   <performance>
856     <properties>
857       <startbeforetolerance>
858         <duration>
859           <duration>T10M</duration>
860         </duration>
861       </startbeforetolerance>
862       <startaftertolerance>
863         <duration>
864           <duration>T0M</duration>
865         </duration>
866       </startaftertolerance>
867       <durationlongtolerance>
868         <duration>
869           <duration>T0M</duration>
870         </duration>
871       </durationlongtolerance>
872       <durationshorttolerance>
873         <duration>
874           <duration>T0M</duration>
875         </duration>
876       </durationshorttolerance>
877     </properties>
878   </performance>
879 </attach>

```

880 In the example, the service can start as much as 10 minutes earlier than the scheduled time, and must
881 start no later than the scheduled time. Whenever the service starts, it the service must execute for exactly
882 the duration indicated.

883 Generally, the implementer should refrain from expressing unnecessary or redundant performance
884 characteristics.

885 4.1.3 Combining Service and Performance

886 Services, references and performance each appear in the ATTACH element of the iCalendar
887 components.

888 *Example 8: Interval with inline XML artifact and optional specified Performance*

```
889 <vtodo>
890   <properties>
891     <x-wscalendartype>Interval</x-wscalendartype>
892     <xcal:uid>
893       <xcal:text>00959BC664CA650E933C892C@example.com</xcal:text>
894     </xcal:uid>
895     <xcal:description>
896       <xcal:text>Sample Contract</xcal:text></xcal:description>
897     <attach>
898       <artifact>
899         <emix-wip>
900           <price>8.45</price>
901           <quantity>8.45</quantityprice>
902         </emix-wip>
903       </artifact/>
904       <performance>
905         <properties>
906           <startbeforetolerance>
907             <duration>
908               <duration>T10M</duration>
909             </duration>
910           </startbeforetolerance>
911           <startaftertolerance>
912             <duration>
913               <duration>T0M</duration>
914             </duration>
915           </startaftertolerance>
916         </properties>
917       </performance>
918     </attach>
919   </properties>
920 </vtodo>
```

921 *Example 9: Interval with external reference and optional specified performance*

```
922 <vtodo>
923   <properties>
924     <x-wscalendartype>Interval</x-wscalendartype>
925     <xcal:uid>
926       <xcal:text>00959BC664CA650E933C892C@example.com</xcal:text>
927     </xcal:uid>
928     <xcal:description>
929       <xcal:text>Sample Contract</xcal:text></xcal:description>
930     <attach>
931       <reference>http://scheduled.ws-calendar-
932         service.com/contract1</reference>
933     </performance>
934     <properties>
935       <startbeforetolerance>
936         <duration>
937           <duration>T10M</duration>
938         </duration>
939       </startbeforetolerance>
940     <startaftertolerance>
```

```
941         <duration>
942             <duration>T0M</duration>
943         </duration>
944     </startaftertolerance>
945 </properties>
946 </performance>
947 </attach>
948 </properties>
949 </vtodo>
```

950

5 Inheritance and Entry Points: Calendar Gluons

951

5.1.1 Calendar Gluons

952

WS-Calendar introduces a new iCalendar component, the Calendar Gluon. In physics, Gluons act to mediate as well as to participate in the interactions between quarks. A Calendar Gluon defines information to be inherited by each Interval in the Sequence, as well as scheduling the entire sequence. A Calendar Gluon is essentially the Interval component profiled down to minimal elements for which inheritance rules are then defined for the sequence. (See Appendix *Overview of WS-Calendar, its Antecedents and its Use*) Calendar Gluons use iCalendar relations to apply service information to Sequences.

957

958

Table 5-1: Calendar Gluon elements in WS-Calendar

Calendar Gluon Element	Use	Discussion
x-wscalendarType	Mandatory xs:string, value always "CalendarGluon"	Added vtodo attribute, ignored by iCalendar processors
dtStamp	[XCAL]:dtstamp <i>Mandatory</i>	Time and date that Calendar Gluon object was created
uid	<i>Mandatory</i>	Used to enable unambiguous referencing of each VTODO object
summary	Text' <i>Optional</i>	Text describing the Calendar Gluon
related	WsCalendar:Relationship <i>Mandatory</i>	A Calendar Gluon must have a relationship with at least one other component. The only relationship defined for the Calendar Gluon is the IsParent.
dtStart	[XCAL]:Time. Start time for the related interval of the sequence. <i>Optional</i>	An Calendar Gluon may either have a dtStart or a dtEnd, but may not have both.
dtEnd	[XCAL]:Time. Scheduled completion time for the related interval of the sequence. <i>Optional</i>	An Calendar Gluon may either have a dtStart or a dtEnd, but may not have both.
duration	[XCAL]:Duration <i>Optional</i>	If specified, a duration is inherited by all intervals in the referred-to sequence,
attach	WSCalendar:Attachment Mandatory Multipleoccurs	Contains WS-Calendar:attachment attribute defining service and performance. Can be inherited by all intervals in sequence.

960 Because the properties of Calendar Gluon properties are inherited by the child Sequence, they can serve
961 as the elements in any Interval in the Sequence. An inherited element can even serve as a substitute for
962 an Interval mandatory element. For example, Duration is mandatory for all Intervals. A Duration
963 expressed in a Calendar Gluon is inherited by each Interval in the associated Sequence. This makes
964 Intervals without internal Duration compliant, because the Interval inherits the Duration from the Calendar
965 Gluon. If an Interval in the associated Sequence does include a Duration, that value overrides the value
966 from the Calendar Gluon.

967 5.1.2 Calendar Gluons and Sequences

968 The Calendar Gluon is used to define common service requirements for an entire sequence. If a
969 RelatedComponent has a parent relationship with the an Interval in a sequence, then the
970 RelatedComponent's Attachment defines service attributes by all Intervals in the Sequence.

971 In this example, the Sequence in the previous example is expressed using an Calendar Gluon.

972 *Example 10: Sequence with Performance defined in the Calendar Gluon*

```
973 <components>  
974 <x- calendargluon>  
975   <properties>  
976     <x-wscalendartype>CalendarGluon</x-wscalendartype>  
977     <xcal:uid>  
978       <xcal:text>00959BC664CA650E933C892C@example.com</xcal:text>  
979     </xcal:uid>  
980     <xcal:description>  
981       <xcal:text> Calendar Gluon with sequence </xcal:text>  
982     </xcal:description>  
983     <xcal:comment>  
984       <xcal:text> creates common performance expectations (+/- 1 second)  
985         for the entire sequence. Also sets common duration (15  
986         minutes) for all members of the sequence, No interval may end  
987         after its scheduled end-time </xcal:text>  
988     </xcal:comment>  
989     <xcal:duration>  
990       <xcal:duration>T15M</xcal:duration>  
991     </xcal:duration>  
992     <attach>  
993       <performance>  
994         <properties>  
995           <endbeforetolerance>  
996             <duration>  
997               <duration>T1S</duration>  
998             </duration>  
999           </endbeforetolerance>  
1000          <endaftertolerance>  
1001            <duration>  
1002              <duration>T1S</duration>  
1003            </duration>  
1004          </endaftertolerance>  
1005          </properties>  
1006        </performance>  
1007      </attach>  
1008      <xcal:related-to>  
1009        <xcal:parameters>  
1010          <xcal:reltype>PARENT</xcal:reltype>  
1011        </xcal:parameters>  
1012        <xcal:text>10959BC664CA650E933C892C@example.com </xcal:text>  
1013      </xcal:related-to>  
1014    </properties>  
1015  </x- calendargluon>  
1016  <vtodo>  
1017    <properties>  
1018      <x-wscalendartype>Interval</x-wscalendartype>
```

```

1019     <xcal:uid>
1020         <xcal:text>10959BC664CA650E933C892C@example.com</xcal:text>
1021     </xcal:uid>
1022     <xcal:description>
1023         <xcal:text>First Interval in Sequence</xcal:text>
1024     </xcal:description>
1025     <xcal:summary>
1026         <xcal:text>Inherits all performance from Gluon as well
1027             As the duration</xcal:text>
1028     </xcal:summary>
1029     <attach>
1030         <artifact>
1031             <emix-wip>
1032                 <price>8.45</price>
1033                 <quantity>4200</quantity>
1034             </emix-wip>
1035         </artifact>
1036     </attach>
1037 </properties>
1038 </vtodo>
1039 <vtodo>
1040     <properties>
1041         <x-wscalendartype>Interval</x-wscalendartype>
1042         <xcal:uid>
1043             <xcal:text>20959BC664CA650E933C892C@example.com</xcal:text>
1044         </xcal:uid>
1045         <xcal:description>
1046             <xcal:text>Second Interval in Sequence</xcal:text>
1047         </xcal:description>
1048         <xcal:summary>
1049             <xcal:text>Inherits all performance from Gluon, follows
1050                 Finish of Interval 1, inherits duration</xcal:text>
1051         </xcal:summary>
1052         <attach>
1053             <artifact>
1054                 <emix-wip>
1055                     <price>8.49</price>
1056                     <quantity>4500</quantity>
1057                 </emix-wip>
1058             </artifact>
1059         </attach>
1060         <x-wscalendarrelation>
1061             <temporalrelationshiptype>finishtostart</temporalrelationshiptype>
1062             <relatedto>
1063                 <uid>10959BC664CA650E933C892C@example.com</uid>
1064             </relatedto>
1065             <gap>
1066                 <xcal:duration>
1067                     <xcal:duration>T0M</xcal:duration>
1068                 </xcal:duration>
1069             </gap>
1070         </x-wscalendarrelation>
1071     </properties>
1072 </vtodo>
1073 <vtodo>
1074     <properties>
1075         <x-wscalendartype>Interval</x-wscalendartype>
1076         <xcal:uid>
1077             <xcal:text>30959BC664CA650E933C892C@example.com</xcal:text>
1078         </xcal:uid>
1079         <xcal:description>
1080             <xcal:text>Third Interval in Sequence</xcal:text>
1081         </xcal:description>
1082     <xcal:summary>

```

```

1083     <xcal:text>Inherits all performance from Gluon, follows
1084         Finish of Interval 2, overrides duration</xcal:text>
1085 </xcal:summary>
1086 <attach>
1087     <artifact>
1088         <emix-wip>
1089             <price>7.45</price>
1090             <quantity>6000</quantity>
1091         </emix-wip>
1092     </artifact>
1093 </attach>
1094 <xcal:duration>
1095     <xcal:duration>T30M</xcal:duration>
1096 </xcal:duration>
1097 <x-wscalendarrelation>
1098     <temporalrelationshipstype>finishtostart</temporalrelationshipstype>
1099     <relatedto>
1100         <uid>20959BC664CA650E933C892C@example.com</uid>
1101     </relatedto>
1102     <gap>
1103         <xcal:duration>
1104             <xcal:duration>T5M</xcal:duration>
1105         </xcal:duration>
1106     </gap>
1107 </x-wscalendarrelation>
1108 </properties>
1109 </vtodo>
1110 </components>

```

1111 Note that the performance expectations, identical for each interval, have moved into the Calendar Gluon.
1112 Not also that while the duration for all intervals in the partition is set in the Calendar Gluon, interval 3
1113 overrides that with a half hour duration assigned locally. The Calendar Gluon happens to related to the
1114 first Interval in the sequence; there are specific use cases (discussed below) which require it to be linked
1115 to other Intervals.

1116 5.1.3 Inheritance rules for Calendar Gluons

1117 In general, the rules that anything specified in the Parent Calendar Gluon applies to each Child. The
1118 Parent of an Interval in a Sequence is parent to all Intervals in the Sequence. As a Sequence creates
1119 single temporal relationship, assigning a start time (dtstart) to any Interval allows the starting time to be
1120 computed for any of them.

1121 *Table 5-2 Gluon Inheritance rules*

Attribute	Inheritance Rules
General	A Interval or Calendar Gluon inherits its attributes through it's the closest parent. Local specification of an attributes overrides any inheritance.
Duration	Follows general rules
Temporal Relation	Relationship Type and Gap only are inherited. Either may be overridden locally. To specify no gap when a parent specifies a gap, an explicit zero duration gap must be specified. Related-to is not inherited.
Performance	Performance is either inherited intact or overridden completely. There are no rules for recombining partial Performance objects through inheritance..

Attribute	Inheritance Rules
Artifacts	Artifacts are combined within their respective namespaces, and are evaluated for completeness after Artifact inheritance. Referring specifications should detail any conformance requirements.
Schedules	In general, schedule dates are inherited as if they consisted of a separate Date and a Time. The Date and the Time are evaluated separately. Thus a child may specify a Date on which it is willing to have a Time scheduled, or a Time at which it is willing to perform a service on any requested Date. If a parent specifies both, and a child specifies one, the paired elements (parent-time:child-time or parent-date:child-date) must match.

1122 5.1.4 Optimizing the expression of a Partition

1123 Partitions are Sequences with consecutive Intervals. Communication of a Partition can be further
 1124 optimized by bringing the relationship into the Calendar Gluon. Notice that while the type of the
 1125 relationship is defined in the Calendar Gluon, the Temporal Relation for each Interval must still be
 1126 expressed within the Interval.

1127 *Example 11: Partition with Duration and Performance defined in the Calendar Gluon*

```

1128 <components>
1129 <x--calendargluon>
1130   <properties>
1131     <x-wscalendartype>CalendarGluon</x-wscalendartype>
1132     <xcal:uid>
1133       <xcal:text>00959BC664CA650E933C892C@example.com</xcal:text>
1134     </xcal:uid>
1135     <xcal:description>
1136       <xcal:text>Calendar Gluon for energy markets with consecutive
1137         Identical intervals</xcal:text>
1138     </xcal:description>
1139     <xcal:comment>
1140       <xcal:text> creates common performance expectations that the
1141         granularity for measuring all Intervals is (+/- 1 second)
1142         Also sets common duration (15 minutes)for all members of
1143         the sequence). Each interval in the partitions begins
1144         immediately after its predecessor finishes. </xcal:text>
1145     </xcal:comment>
1146     <xcal:duration>
1147       <xcal:duration>T15M</xcal:duration>
1148     </xcal:duration>
1149     <attach>
1150       <artifact>
1151         <emix-wip>PRODUCT SPECIFICATION UNDEFINED</emix-wip>
1152       </artifact/>
1153       <performance>
1154         <properties>
1155           <granularity>
1156             <duration>
1157               <duration>T1S</duration>
1158             </duration>
1159           </granularity>
1160         </properties>
1161       </performance>
1162     </attach>
1163     <x-wscalendarrelation>
1164       <temporalrelationshipiptye>finishtostart</temporalrelationshipiptye>
1165     <gap>

```

```

1166         <xcal:duration>
1167             <xcal:duration>T0S</xcal:duration>
1168         </xcal:duration>
1169     </gap>
1170 </x-wscalendarrelation>
1171 <xcal:related-to>
1172     <xcal:parameters>
1173         <xcal:reltype>PARENT</xcal:reltype>
1174     </xcal:parameters>
1175     <xcal:text>10959BC664CA650E933C892C@example.com </xcal:text>
1176 </xcal:related-to>
1177 </properties>
1178 </x-calendargluon>
1179 <vtodo>
1180     <properties>
1181         <x-wscalendartype>Interval</x-wscalendartype>
1182         <xcal:uid>
1183             <xcal:text>10959BC664CA650E933C892C@example.com</xcal:text>
1184         </xcal:uid>
1185         <xcal:description>
1186             <xcal:text>First Interval in Sequence</xcal:text>
1187         </xcal:description>
1188         <xcal:summary>
1189             <xcal:text>Inherits all performance from Gluon as well
1190                 As the duration and the Temporal Relation</xcal:text>
1191         </xcal:summary>
1192         <attach>
1193             <artifact>
1194                 <emix-wip>
1195                     <price>8.45</price>
1196                     <quantity>4200</quantity>
1197                 </emix-wip>
1198             </artifact/>
1199         </attach>
1200     </properties>
1201 </vtodo>
1202 <vtodo>
1203     <properties>
1204         <x-wscalendartype>Interval</x-wscalendartype>
1205         <xcal:uid>
1206             <xcal:text>20959BC664CA650E933C892C@example.com</xcal:text>
1207         </xcal:uid>
1208         <xcal:description>
1209             <xcal:text>Second Interval in Sequence</xcal:text>
1210         </xcal:description>
1211         <attach>
1212             <artifact>
1213                 <emix-wip>
1214                     <price>8.49</price>
1215                     <quantity>4500</quantity>
1216                 </emix-wip>
1217             </artifact/>
1218         </attach>
1219         <x-wscalendarrelation>
1220             <relatedto>
1221                 <uid>10959BC664CA650E933C892C@example.com</uid>
1222             </relatedto>
1223         </x-wscalendarrelation>
1224     </properties>
1225 </vtodo>
1226 <vtodo>
1227     <properties>
1228         <x-wscalendartype>Interval</x-wscalendartype>
1229         <xcal:uid>

```



```

1230     <xcal:text>30959BC664CA650E933C892C@example.com</xcal:text>
1231 </xcal:uid>
1232 <xcal:description>
1233     <xcal:text>Third Interval in Sequence</xcal:text>
1234 </xcal:description>
1235 <xcal:summary>
1236     <xcal:text>Inherits all performance from Gluon, follows
1237     Finish of Interval 2, overrides duration</xcal:text>
1238 </xcal:summary>
1239 <attach>
1240     <artifact>
1241         <emix-wip>
1242             <price>7.45</price>
1243             <quantity>6000</quantity>
1244         </emix-wip>
1245     </artifact/>
1246 </attach>
1247 <x-wscalendarrelation>
1248     <relatedto>
1249         <uid>20959BC664CA650E933C892C@example.com</uid>
1250     </relatedto>
1251 </x-wscalendarrelation>
1252 </properties>
1253 </vtodo>
1254 </components>

```

1255 This Partition shows a school schedule in which classes start one hour apart. Each service is performed
1256 for 50 minutes, and there is a 10 minute gap between each as students move between classes. Classes
1257 may not begin before the schedule, but they may start up to five minutes late.

1258 Stripped of all annotations, this can be expressed as follows:

1259 *Example 12: Partition without annotations*

```

1260 <components>
1261 <x-calendargluon>
1262     <properties>
1263         <x-wscalendarstype>CalendarGluon</x-wscalendarstype>
1264         <xcal:uid>
1265             <xcal:text>00959BC664CA650E933C892C@example.com</xcal:text>
1266         </xcal:uid>
1267         <xcal:duration>
1268             <xcal:duration>T50M</xcal:duration>
1269         </xcal:duration>
1270         <attach>
1271             <artifact>
1272                 <classroom>demonstration specification</classroom>
1273             </artifact/>
1274         </attach>
1275         <x-wscalendarrelation>
1276             <temporalrelationshipstype>finishtostart</temporalrelationshipstype>
1277             <gap>
1278                 <xcal:duration>
1279                     <xcal:duration>T10M</xcal:duration>
1280                 </xcal:duration>
1281             </gap>
1282         </x-wscalendarrelation>
1283         <xcal:related-to>
1284             <xcal:parameters>
1285                 <xcal:reltype>PARENT</xcal:reltype>
1286             </xcal:parameters>
1287             <xcal:text>10959BC664CA650E933C892C@example.com </xcal:text>
1288         </xcal:related-to>
1289     </properties>
1290 </x-calendargluon>

```

```

1291 <vtodo>
1292   <properties>
1293     <x-wscalendartype>Interval</x-wscalendartype>
1294     <xcal:uid>
1295       <xcal:text>10959BC664CA650E933C892C@example.com</xcal:text>
1296     </xcal:uid>
1297     <attach>
1298       <artifact>
1299         <classroom><students>48</students></classroom>
1300       <artifact/>
1301     </attach>
1302   </properties>
1303 </vtodo>
1304 <vtodo>
1305   <properties>
1306     <x-wscalendartype>Interval</x-wscalendartype>
1307     <xcal:uid>
1308       <xcal:text>20959BC664CA650E933C892C@example.com</xcal:text>
1309     </xcal:uid>
1310     <attach>
1311       <artifact>
1312         <classroom><students>65</students></classroom>
1313       <artifact/>
1314     </attach>
1315     <x-wscalendarrelation>
1316       <relatedto>
1317         <uid>10959BC664CA650E933C892C@example.com</uid>
1318       </relatedto>
1319     </x-wscalendarrelation>
1320   </properties>
1321 </vtodo>
1322 <vtodo>
1323   <properties>
1324     <x-wscalendartype>Interval</x-wscalendartype>
1325     <xcal:uid>
1326       <xcal:text>30959BC664CA650E933C892C@example.com</xcal:text>
1327     </xcal:uid>
1328     <attach>
1329       <artifact>
1330         <classroom><students>34</students></classroom>
1331       <artifact/>
1332     </attach>
1333     <x-wscalendarrelation>
1334       <relatedto>
1335         <uid>20959BC664CA650E933C892C@example.com</uid>
1336       </relatedto>
1337     </x-wscalendarrelation>
1338   </properties>
1339 </vtodo>
1340 <components>

```

1341 A sequence can also be scheduled in the Calendar Gluon.

1342 *Example 13: A Scheduled Sequence showing Temporal Relationship Inheritance*

```

1343 <components>
1344 <x-calendargluon>
1345   <properties>
1346     <x-wscalendartype>CalendarGluon</x-wscalendartype>
1347     <xcal:uid>
1348       <xcal:text>00959BC664CA650E933C892C@example.com</xcal:text>
1349     </xcal:uid>
1350     <dtstart>2010-09-11 T00:15</dtstart>
1351     <attach>
1352       <artifact>

```

```

1353         <classroom>demonstration specification</classroom>
1354         <artifact/>
1355     </attach>
1356     <xcal:related-to>
1357         <xcal:parameters>
1358             <xcal:reltype>PARENT</xcal:reltype>
1359         </xcal:parameters>
1360         <xcal:text>10959BC664CA650E933C892C@example.com </xcal:text>
1361     </xcal:related-to>
1362 </properties>
1363 </x-calendargluon>
1364 <vtodo>
1365     <properties>
1366         <x-wscalendartype>Interval</x-wscalendartype>
1367         <xcal:uid>
1368             <xcal:text>10959BC664CA650E933C892C@example.com</xcal:text>
1369         </xcal:uid>
1370         <xcal:description>
1371             <xcal:text>First Interval in Sequence</xcal:text>
1372         </xcal:description>
1373         <dtstart>2010-09-11T13:00</dtstart>
1374         <xcal:duration>
1375             <xcal:duration>T1H</xcal:duration>
1376         </xcal:duration>
1377     </properties>
1378 </vtodo>
1379 <vtodo>
1380     <properties>
1381         <x-wscalendartype>Interval</x-wscalendartype>
1382         <xcal:uid>
1383             <xcal:text>20959BC664CA650E933C892C@example.com</xcal:text>
1384         </xcal:uid>
1385         <xcal:description>
1386             <xcal:text>Second Interval in Sequence</xcal:text>
1387         </xcal:description>
1388         <xcal:duration>
1389             <xcal:duration>T15M</xcal:duration>
1390         </xcal:duration>
1391         <x-wscalendarrelation>
1392             <temporalrelationshipi>finishtostart</temporalrelationshipi>
1393             <relatedto>
1394                 <uid>10959BC664CA650E933C892C@example.com</uid>
1395             </relatedto>
1396         </x-wscalendarrelation>
1397     </properties>
1398 </vtodo>
1399 </components>

```

1400 5.1.5 Mixed Inheritance of Start Time

1401 A Sequence has not been scheduled until it has both a start time and a start date. Start time and date
1402 SHALL be expressed together when all components are in a single communication. Time and Date MAY
1403 be separated when the full sequence and schedule are created by reference.

1404 To illustrate this, here is the classroom scheduling Partition from Example 12, updated to include each
1405 day's school opening.

1406 *Example 14: Partition with Duration and Performance defined in the Calendar Gluon*

```

1407 <components>
1408 <x-calendargluon>
1409     <properties>
1410         <x-wscalendartype>CalendarGluon</x-wscalendartype>
1411         <xcal:uid>

```

```

1412         <xcal:text>
1413             90959BC664CA650E933C892C@invokingexample.com
1414         </xcal:text>
1415     </xcal:uid>
1416     <dtstart>2010-09-13T09:00</dtstart>
1417     <xcal:related-to>
1418         <xcal:parameters>
1419             <xcal:reltype>PARENT</xcal:reltype>
1420         </xcal:parameters>
1421         <xcal:uri>http://scheduled.ws-calendar-service.com/classSchedule
1422             </xcal:uri>
1423     </xcal:related-to>
1424 </properties>
1425 </x-calendargluon>
1426 </components>

```

1427 Here, an external Calendar Gluon (above) makes reference to a published classroom schedule service:

```

1428 <x-calendargluon>
1429     <properties>
1430         <x-wscalendartype>CalendarGluon</x-wscalendartype>
1431         <xcal:uid>
1432             <xcal:text>http://scheduled.ws-calendar-service.com/classSchedule
1433             </xcal:text>
1434         </xcal:uid>
1435         <xcal:description>
1436             <xcal:text>MWF Classroom Schedule
1437                 Identical intervals</xcal:text>
1438         </xcal:description>
1439         <xcal:comment>
1440             <xcal:text>Publishes a common classroom schedule for Monday,
1441                 Wednesday, Friday for a semester at a school. Note that each
1442                 day starts at 9:00</xcal:text>
1443         </xcal:comment>
1444         <dtstart>T09:00</dtstart>
1445         <xcal:duration>
1446             <xcal:duration>T50M</xcal:duration>
1447         </xcal:duration>
1448         <attach>
1449             <xcal:uri></xcal:uri>
1450         </attach>
1451         <x-wscalendarrelation>
1452             <temporalrelationshipi>type>finishtostart</temporalrelationshipi>
1453             <gap>
1454                 <xcal:duration>
1455                     <xcal:duration>T10M</xcal:duration>
1456                 </xcal:duration>
1457             </gap>
1458         </x-wscalendarrelation>
1459         <xcal:related-to>
1460             <xcal:parameters>
1461                 <xcal:reltype>PARENT</xcal:reltype>
1462             </xcal:parameters>
1463             <xcal:text>10959BC664CA650E933C892C@example.com </xcal:text>
1464         </xcal:related-to>
1465     </properties>
1466 </x-calendargluon>
1467 <vtodo>
1468     <properties>
1469         <x-wscalendartype>Interval</x-wscalendartype>
1470         <xcal:uid>
1471             <xcal:text>10959BC664CA650E933C892C@example.com</xcal:text>
1472         </xcal:uid>
1473         <attach>

```

```

1474         <artifact>
1475             <classroom><students>48</students></classroom>
1476         </artifact>
1477     </attach>
1478 </properties>
1479 </vtodo>
1480 <vtodo>
1481     <properties>
1482         <x-wscalendartype>Interval</x-wscalendartype>
1483         <xcal:uid>
1484             <xcal:text>20959BC664CA650E933C892C@example.com</xcal:text>
1485         </xcal:uid>
1486         <attach>
1487             <artifact>
1488                 <classroom><students>65</students></classroom>
1489             </artifact>
1490         </attach>
1491         <x-wscalendarrelation>
1492             <relatedto>
1493                 <uid>10959BC664CA650E933C892C@example.com</uid>
1494             </relatedto>
1495         </x-wscalendarrelation>
1496     </properties>
1497 </vtodo>
1498 <vtodo>
1499     <properties>
1500         <x-wscalendartype>Interval</x-wscalendartype>
1501         <xcal:uid>
1502             <xcal:text>30959BC664CA650E933C892C@example.com</xcal:text>
1503         </xcal:uid>
1504         <attach>
1505             <artifact>
1506                 <classroom><students>34</students></classroom>
1507             </artifact>
1508         </attach>
1509         <x-wscalendarrelation>
1510             <relatedto>
1511                 <uid>20959BC664CA650E933C892C@example.com</uid>
1512             </relatedto>
1513         </x-wscalendarrelation>
1514     </properties>
1515 </vtodo>
1516 <components>
1517

```

1518 In the example above, a general purpose classroom calendar has been created and advertised with an
1519 URL. The class day always starts at 9:00. The referring Calendar Gluon scheduled a particular instance
1520 of this Sequence for Monday, September 13.

1521 This double inheritance, in which a Sequence inherits from a Calendar Gluon which inherits from a
1522 Calendar Gluon is a useful pattern for scheduling an advertised service.

1523 5.1.6 Other Scheduling Scenarios

1524 Sometimes, the invoker of a service is interested only in single Interval of the Sequence, but the entire
1525 Sequence is required. In the example below, the second Interval is advertised, i.e., the Calendar Gluon
1526 points to the second Interval. The first interval might be a required ramp-period, during which the
1527 underlying process is “warming up”, and which may bring some lesser service to market during that ramp
1528 time. The ramp-down time at the end is similarly fixed. The entire Service offering is represented by the
1529 exposed (it has a public URI) Calendar Gluon.

1530 *Example 15: Standard Sequence with Ramp-Up and Ramp Down*

1531 <components>

```

1532 <x- calendargluon>
1533   <properties>
1534     <x-wscalendartype>CalendarGluon</x-wscalendartype>
1535     <xcal:uid>
1536       <xcal:text>http://scheduled.ws-calendar-service.com/runcontract
1537       </xcal:text>
1538     </xcal:uid>
1539     <xcal:description>
1540       <xcal:text>Advertisement of schedule with ramp up and ramp down
1541       services.</xcal:text>
1542     </xcal:description>
1543     <xcal:comment>
1544       <xcal:text>Invokes second of three Intervals in the Sequence
1545       </xcal:text>
1546     </xcal:comment>
1547     <attach>
1548       <xcal:uri><emix-wip4></emix-wip></xcal:uri>
1549     </attach>
1550     <xcal:related-to>
1551       <xcal:parameters>
1552         <xcal:reltype>PARENT</xcal:reltype>
1553       </xcal:parameters>
1554       <xcal:text>20959BC664CA650E933C892C@example.com </xcal:text>
1555     </xcal:related-to>
1556   </properties>
1557   <xcal:duration>
1558     <xcal:duration>T6H</xcal:duration>
1559   </xcal:duration>
1560 </x- calendargluon>
1561 <vtodo>
1562   <properties>
1563     <x-wscalendartype>Interval</x-wscalendartype>
1564     <xcal:uid>
1565       <xcal:text>10959BC664CA650E933C892C@example.com</xcal:text>
1566     </xcal:uid>
1567     <xcal:description>
1568       <xcal:text>Ramp-Up Interval</xcal:text>
1569     </xcal:description>
1570     <xcal:summary>
1571       <xcal:text>Required as part of operations. Slowly increasing, yet
1572       fixed over-all, energy produced.
1573     </xcal:text>
1574     </xcal:summary>
1575     <xcal:duration>
1576       <xcal:duration>T45M</xcal:duration>
1577     </xcal:duration>
1578     <attach>
1579       <artifact>
1580         <emix-wip>describes ramp-up</emix-wip>
1581       </artifact>
1582     </attach>
1583   </properties>
1584 </vtodo>
1585 <vtodo>
1586   <properties>
1587     <x-wscalendartype>Interval</x-wscalendartype>
1588     <xcal:uid>
1589       <xcal:text>20959BC664CA650E933C892C@example.com</xcal:text>
1590     </xcal:uid>
1591     <xcal:description>

```

⁴ There is no EMIX-WIP specification. EMIX-WIP represents a generic energy market artifact, perhaps indicative of future specifications.

```

1592         <xcal:text>Run Interval</xcal:text>
1593     </xcal:description>
1594     <xcal:summary>
1595         <xcal:text>Inherits all performance from Gluon, follows
1596             Finish of Interval 2, overrides duration</xcal:text>
1597     </xcal:summary>
1598     <attach>
1599         <artifact>
1600             <emix-wip >Product Definition</emix-wip>
1601         </artifact>
1602     </attach>
1603     <x-wscalendarrelation>
1604         <relatedto>
1605             <uid>10959BC664CA650E933C892C@example.com</uid>
1606         </relatedto>
1607     </x-wscalendarrelation>
1608 </properties>
1609 </vtodo>
1610 <vtodo>
1611     <properties>
1612         <x-wscaleardtype>Interval</x-wscaleardtype>
1613         <xcal:uid>
1614             <xcal:text>30959BC664CA650E933C892C@example.com</xcal:text>
1615         </xcal:uid>
1616         <xcal:description>
1617             <xcal:text>Ramp-down Interval</xcal:text>
1618         </xcal:description>
1619         <xcal:summary>
1620             <xcal:text>Required as part of operations. Fixed time and fixed,
1621                 while diminishing, energy produced.
1622             </xcal:text>
1623         </xcal:summary>
1624         <xcal:duration>
1625             <xcal:duration>T30M</xcal:duration>
1626         </xcal:duration>
1627         <attach>
1628             <artifact>
1629                 <emix-wip>describes ramp-up</emix-wip>
1630             </artifact>
1631         </attach>
1632         <x-wscalendarrelation>
1633             <relatedto>
1634                 <uid>20959BC664CA650E933C892C@example.com</uid>
1635             </relatedto>
1636         </x-wscalendarrelation>
1637     </properties>
1638 </vtodo>

```

1639 When the service is scheduled, the time and duration are specified. The duration only applies to the
1640 Second Interval as all others have their duration explicitly specified.

```

1641 <components>
1642 <x-calendargluon>
1643     <properties>
1644         <x-wscaleardtype>CalendarGluon</x-wscaleardtype>
1645         <xcal:uid>
1646             <xcal:text>http://scheduled.ws-calendar-service.com/scheduleB
1647             </xcal:text>
1648         </xcal:uid>
1649         <xcal:description>
1650             <xcal:text>Advertisement of schedule with ramp up and ramp down
1651                 services.</xcal:text>
1652         </xcal:description>
1653         <xcal:comment>

```

```
1654         <xcal:text>Invokes second of three Intervals in the Sequence
1655         </xcal:text>
1656     </xcal:comment>
1657     <attach>
1658         <artifact>
1659             <emix-wip5>
1660                 <execute-price>15000</execute-price>
1661             </emix-wip>
1662         </artifact>
1663     </attach>
1664     <xcal:related-to>
1665         <xcal:parameters>
1666             <xcal:reltype>PARENT</xcal:reltype>
1667         </xcal:parameters>
1668         <xcal:text>http://scheduled.ws-calendar-service.com/runcontract
1669         </xcal:text>
1670     </xcal:related-to>
1671     <xcal:duration>
1672         <xcal:duration>T6H</xcal:duration>
1673     </xcal:duration>
1674 </properties>
1675 </x-calendargluon>
1676 </components>
```

1677 In this case, the specific interval is scheduled and a run time of 6 hours is specified for a price of \$15,000.

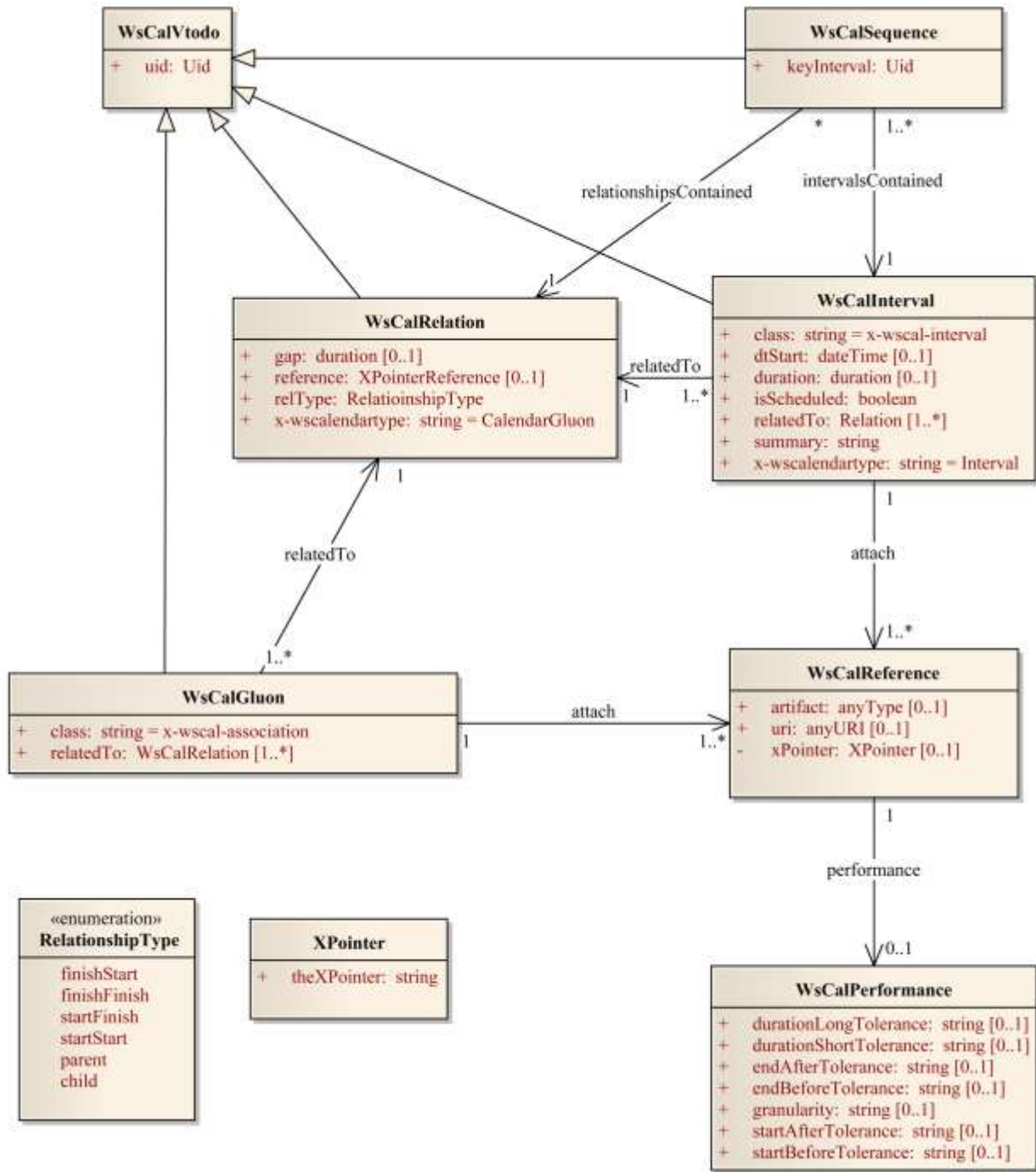
⁵ There is no EMIX-WIP specification. EMIX-WIP represents a generic energy market artifact, perhaps evocative of future specifications.

1678

6 WS-Calendar Models

1679

6.1 Abstract model for WS-Calendar Objects



1680

1681

1682

Figure 1: Abstract UML Model

6.2 Implementation Model for WS-Calendar

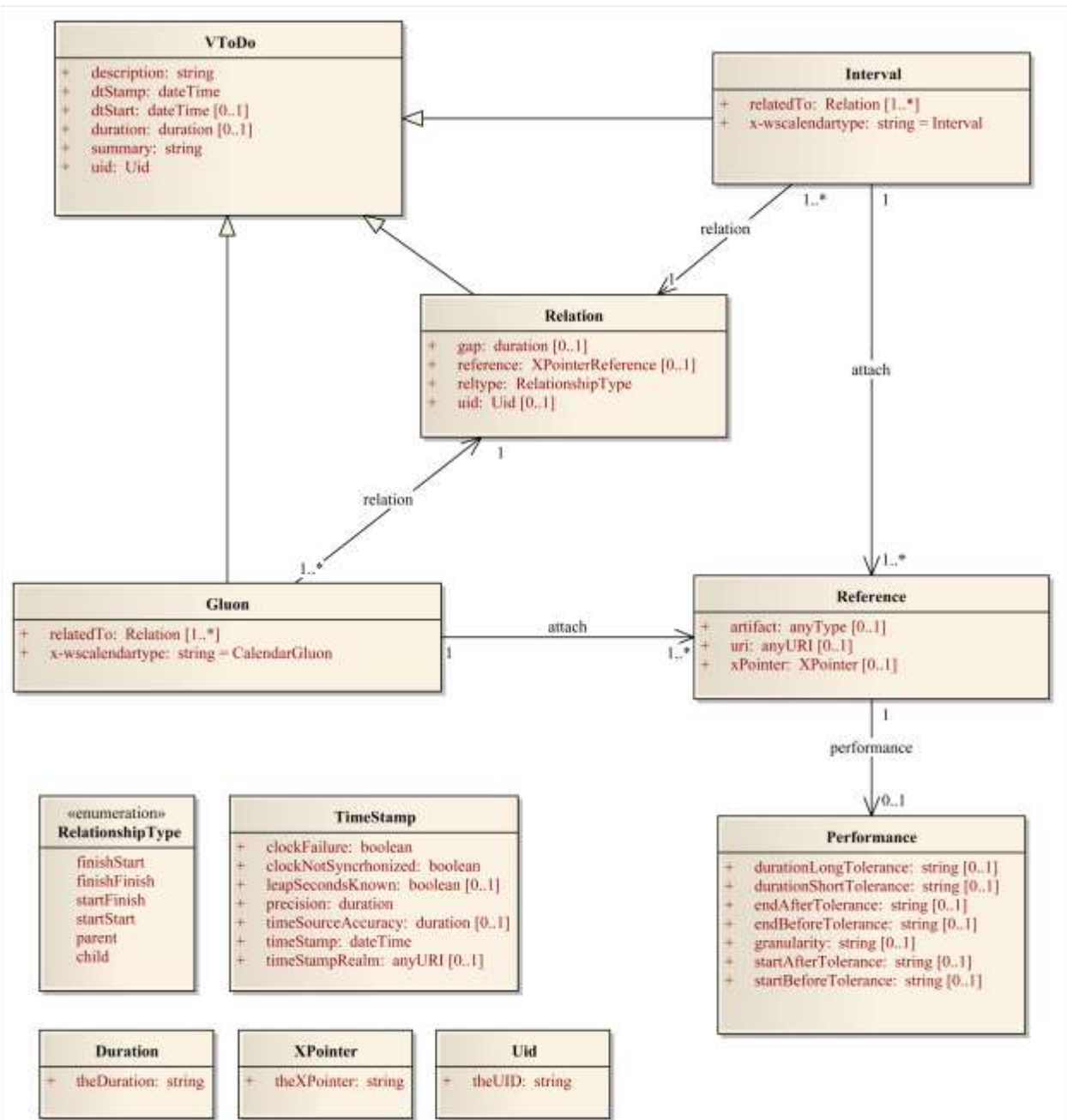


Figure 2: Implementation Model for WS-Calendar

1686 7 Calendar Services

1687 The Service interactions are built upon and make the same assumptions about structure as the CalDAV
1688 protocol defined in **[RFC4791]** and related specifications. It does NOT require nor assume the WebDAV
1689 nor CalDAV protocol but does make use of some of the same elements and structures in the CalDAV
1690 XML namespace.

1691 Calendar resources, for example events and tasks are stored as named resources (files) inside special
1692 collections (folders) known as "**Calendar Collections**".

1693 These services can be looked upon as a layer built on top of CalDAV and defines the basic operations
1694 which allow creation, retrieval, update and deletion. In addition, query, and free-busy operations are
1695 defined to allow efficient, partial retrieval of calendar data.

1696 These services assume a degree of conformity with CalDAV is established such that services built in that
1697 manner do not have a significant mismatch. It is assumed that some WS-Calendar services will be built
1698 without any CalDAV support.

1699 7.1 Overview of the protocol

1700 The protocol is an HTTP based RESTfull protocol using a limited set of methods. Each request may be
1701 followed by a response containing status information.

1702 The following methods are specified in the protocol description, PUT, POST, GET, DELETE. To avoid
1703 various issues with certain methods being blocked clients may use the X-HTTP-Method-Override: header
1704 to specify the intended operation. Servers SHOULD behave as if the named method was used.

```
1705 POST /user/fred/calendar/ HTTP/1.1  
1706 ...  
1707 X-HTTP-Method-Override: PUT  
1708 Properties
```

1709 A service or resource will have a number of properties which describe the current state of that service or
1710 resource. These properties are accessed through a GET on the target resource or service with an
1711 ACCEPT header specifying application/xrd+xml. See Section 7.1.3.6

1712 The following operations are defined by this specification:

- 1713 • Retrieval and update of service and resource properties
- 1714 • Creation of a calendar object
- 1715 • Retrieval of a calendar object
- 1716 • Update of a calendar object
- 1717 • Deletion of a calendar object
- 1718 • Query
- 1719 • Free-busy query

1720 7.1.1 Calendar Object Resources

1721 The same restrictions apply to Calendar Object Resources as specified in CalDAV **[RFC4791]** section
1722 4.2. An additional constraint for CalWS is that no timezone specifications are transferred.

1723 7.1.2 Timezone information

1724 It is assumed that the client and server each have access to a full set of up to date timezone information.
1725 Timezones will be referenced by a timezone identifier from the full set of Olson data together with a set of
1726 well-known aliases defined **[TZDB]**. CalWS services may advertise themselves as timezone servers
1727 through the server properties object.

1728 **7.1.3 Issues not addressed by this specification.**

1729 A number of issues are not addressed by this version of the specification, either because they should be
1730 addressed elsewhere or will be addressed at some later date.

1731 **7.1.3.1 Access Control**

1732 It is assumed that the targeted server will set an appropriate level of access based on authentication. This
1733 specification will not attempt to address the issues of sharing or Access Control Lists (ACLs).

1734 **7.1.3.2 Provisioning**

1735 The protocol will not provide any explicit provisioning operations. If it is possible to authenticate or
1736 address a principals calendar resources then they **MUST** be automatically created if necessary or
1737 appropriate

1738 **7.1.3.3 Copy/Move**

1739 These operations are not yet defined for this version of the CalWS protocol. Both operations raise a
1740 number of issues. In particular implementing a move operation through a series of retrievals, insertions
1741 and deletions may cause undesirable side-effects. Both these operations will be defined in a later version
1742 of this specification.

1743 **7.1.3.4 Creating Collections**

1744 We will not address the issue of creating collections within the address space. The initial set is created by
1745 provisioning.

1746 **7.1.3.5 Retrieving collections**

1747 This operation is currently undefined. A GET on a collection may fail or return a complete calendar object
1748 representing the collection.

1749 **7.1.3.6 Setting service and resource properties.**

1750 These operations are not defined in this version of the specification. In the future it will be possible to
1751 define or set the properties for the service or resources within the service.

1752 **7.1.4 CalWS Glossary**

1753 **7.1.4.1 Hrefs**

1754 An href is a URI reference to a resource, for example

1755 `"http://example.org/user/fred/calendar/event1.ics".`

1756 The URL above reflects a possible structure for a calendar server. All URLs should be absolute or path-
1757 absolute following the rules defined in **RFC4918** Section 8.3.

1758 **7.1.4.2 Calendar Object Resource**

1759 A calendar object resource is an event, meeting or a task. Attachments are resources but **NOT** calendar
1760 object resources. An event or task with overrides is a single calendar resource entity.

1761 **7.1.4.3 Calendar Collection**

1762 A folder only allowed to contain calendar object resources.

1763 7.1.4.4 Scheduling Calendar Collection

1764 A folder only allowed to contain calendar resources which is also used for scheduling operations.
1765 Scheduling events placed in such a collection will trigger implicit scheduling activity on the server.

1766 7.1.4.5 Principal Home

1767 The collection under which all the resources for a given principal are stored. For example, for principal
1768 "fred" the principal home might be "/user/fred/"

1769 7.2 Error conditions

1770 Each operation on the calendar system has a number of pre-conditions and post-conditions that apply.

1771 A "precondition" for a method describes the state of the server that must be true for that method to be
1772 performed. A "post-condition" of a method describes the state of the server that must be true after that
1773 method has been completed. Any violation of these conditions will result in an error response in the form
1774 of a CalWS XML error element containing the violated condition and an optional description. \

1775 Each method specification defines the preconditions that must be satisfied before the method can
1776 succeed. A number of post-conditions are generally specified which define the state that must exist after
1777 the execution of the operation. Preconditions and post-conditions are defined as error elements in the
1778 CalWS XML namespace.

1779 7.2.1 Example: error with CalDAV error condition

```
1780 <?xml version="1.0" encoding="utf-8"  
1781     xmlns:CW="Error! Reference source not found."  
1782     xmlns:C="urn:ietf:params:xml:ns:caldav" ?>  
1783 <CW:error>  
1784   <C:supported-filter>  
1785     <C:prop-filter name="X-ABC-GUID"/>  
1786   </C:supported-filter>  
1787   <CW:description>Unknown property </CW:description>  
1788 </CW:error>
```

1789

8 Properties and link relations

1790

8.1 Property and relation-type URIs

1791
1792
1793
1794

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

1795
1796
1797

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

1798

```
http://docs.oasis-open.org/ns/wscal/calws/caldav/supported-calendar-data
```

1799

corresponds to

1800

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

1801
1802

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

1803

8.2 supported-features property.

1804

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

1805
1806
1807

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

1808

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

1809
1810

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

1811

8.3 max-attendees-per-instance

1812

```
http://docs.oasis-open.org/ns/wscal/calws/max-attendees-per-instance
```

1813

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

1814

8.4 max-date-time

1815

```
http://docs.oasis-open.org/ns/wscal/calws/max-date-time
```

1816

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

1817

8.5 max-instances

1818

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

1819

Defines the maximum number of instances allowed per event or task

1820

8.6 max-resource-size

1821

```
http://docs.oasis-open.org/ns/wscal/calws/max-resource-size
```

1822
1823

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

1824 **8.7 min-date-time**

1825 <http://docs.oasis-open.org/ns/wscal/calws/min-date-time>

1826 Provides a DATE-TIME value indicating the earliest date and time (in UTC) that the server is willing to
1827 accept for any DATE or DATE-TIME value in a calendar object resource stored in a calendar collection.

1828 **8.8 description**

1829 <http://docs.oasis-open.org/ns/wscal/calws/description>

1830 Provides some descriptive text for the targeted collection.

1831 **8.9 timezone-service relation.**

1832 <http://docs.oasis-open.org/ns/wscal/calws/timezone-service>

1833 The location of a timezone service used to retrieve timezone information and specifications. This may be
1834 an absolute URL referencing some other service or a relative URL if the current server also provides a
1835 timezone service.

```
1836 <Link rel="http://docs.oasis-open.org/ns/wscal/calws/calws/timezone-service"  
1837 href="http://example.com/tz" />
```

1838 **8.10 principal-home relation.**

1839 <http://docs.oasis-open.org/ns/wscal/calws/principal-home>

1840 Provides the URL to the user home for the currently authenticated principal.

```
1841 <Link rel="http://docs.oasis-open.org/ns/wscal/calws/principal-home"  
1842 href="http://example.com/user/fred" />
```

1843 **8.11 current-principal-freebusy relation.**

1844 <http://docs.oasis-open.org/ns/wscal/calws/current-principal-freebusy>

1845 Provides the URL to use as a target for freebusy requests for the current authenticated principal.

```
1846 <Link rel="http://docs.oasis-open.org/ns/wscal/calws/current-principal-freebusy"  
1847 href="http://example.com/freebusy/user/fred" />
```

1848 **8.12 principal-freebusy relation.**

1849 <http://docs.oasis-open.org/ns/wscal/calws/principal-freebusy>

1850 Provides the URL to use as a target for freebusy requests for a different principal.

```
1851 <Link rel="http://docs.oasis-open.org/ns/wscal/calws/principal-freebusy"  
1852 href="http://example.com/freebusy" />
```

1853 **8.13 child-collection relation.**

1854 <http://docs.oasis-open.org/ns/wscal/calws/child-collection>

1855 Provides information about a child collections for the target. The href attribute gives the URI of the
1856 collection. The element should only have CalWS child elements giving the type of the collection, that is
1857 the CalWS:collection link property and the CalWS-calendar-collection link property. This allows clients to
1858 determine the structure of a hierarchical system by targeting each of the child collections in turn.

1859 The xrd:title child element of the link element provides a description for the child-collection.

```
1860 <Link rel="http://http://docs.oasis-open.org/ns/wscal/calws/child-collection"  
1861 href="http://example.com/calws/user/fred/calendar">  
1862 <Title xml:lang="en">Calendar</Title>  
1863 <Property type="http://docs.oasis-open.org/ns/wscal/calws/collection"  
1864 xsi:nil="true" />
```


1865 <Property type="http://docs.oasis-open.org/ns/wscal/calws/calendar-
1866 collection"
1867 xsi:nil="true" />
1868 </Link>

1869 **8.14 created link property**

1870 <http://docs.oasis-open.org/ns/wscal/calws/created>

1871 Appears within a link relation describing collections or entities. The value is a date-time as defined in
1872 RFC3339 Section 5.6

1873 <Property type="http://docs.oasis-open.org/ns/wscal/calws/created"
1874 >1985-04-12T23:20:50.52Z</Property>

1875 **8.15 last-modified property**

1876 <http://docs.oasis-open.org/ns/wscal/calws/last-modified>

1877 Appears within an xrd object describing collections or entities. The value is the same format as would
1878 appear in the Last-Modified header and is defined in **[RFC2616]**, Section 3.3.1

1879 <Property type="http://docs.oasis-open.org/ns/wscal/calws/last-modified"
1880 >Mon, 12 Jan 1998 09:25:56 GMT</Property>

1881 **8.16 displayname property**

1882 <http://docs.oasis-open.org/ns/wscal/calws/displayname>

1883 Appears within an xrd object describing collections or entities. The value is a localized name for the entity
1884 or collection.

1885 <Property type="http://docs.oasis-open.org/ns/wscal/calws/displayname"
1886 >My Calendar</Property>

1887 **8.17 timezone property**

1888 <http://docs.oasis-open.org/ns/wscal/calws/timezone>

1889 Appears within an xrd object describing collections. The value is a text timezone identifier.

1890 <Property type="http://docs.oasis-open.org/ns/wscal/calws/timezone"
1891 >America/New_York</Property>

1892 **8.18 owner property**

1893 <http://docs.oasis-open.org/ns/wscal/calws/owner>

1894 Appears within an xrd object describing collections or entities. The value is a server specific uri.

1895 <Property type="http://docs.oasis-open.org/ns/wscal/calws/owner"
1896 >/principals/users/mike</Property>

1897 **8.19 collection link property**

1898 <http://docs.oasis-open.org/ns/wscal/calws/collection>

1899 Appears within a link relation describing collections or entities. The property takes no value and indicates
1900 that this child element is a collection.

1901 <Property type="http://docs.oasis-open.org/ns/wscal/calws/collection"
1902 xsi:nil="true" />

1903 **8.20 calendar-collection link property**

1904 <http://docs.oasis-open.org/ns/wscal/calws/calendar-collection>

1905 Appears within a link relation describing collections or entities. The property takes no value and indicates
1906 that this child element is a calendar collection.

```
1907 <Property type="http://docs.oasis-open.org/ns/wscal/calws/calendar-collection"  
1908 xsi:nil="true" />
```

1909 **8.21 CalWS:privilege-set XML element**

1910 <http://docs.oasis-open.org/ns/wscal/calws:privilege-set>

1911 Appears within a link relation describing collections or entities and specifies the set of privileges allowed
1912 to the current authenticated principal for that collection or entity.

```
1913 <!ELEMENT calws:privilege-set (calws:privilege*)>  
1914 <!ELEMENT calws:privilege ANY>
```

1915 Each privilege element defines a privilege or access right. The following set is currently defined

- 1916 • CalWS: Read - current principal has read access
- 1917 • CalWS: Write - current principal has write access

```
1918 <calws:privilege-set>  
1919 <calws:privilege><calws:read></calws:privilege>  
1920 <calws:privilege><calws:write></calws:privilege>  
1921 </calws:privilege-set>
```

1922

9 Retrieving Collection and Service Properties

1923 Properties, related services and locations are obtained from the service or from service resources in the
1924 form of an XRD document as defined by [XRD-1.0].

1925 Given the URL of a CalWS service a client retrieves the service XRD document through a GET on the
1926 service URL with an ACCEPT header specifying application/xrd+xml.

1927 Retrieving resource properties is identical to obtaining service properties, that is, execute a GET on the
1928 target URL with an ACCEPT header specifying application/xrd+xml.

1929 The service properties define the global limits and defaults. Any properties defined on collections within
1930 the service hierarchy override those service defaults. The service may choose to prevent such overriding
1931 of defaults and limits when appropriate.

1932 9.1 Request parameters

- 1933 • None

1934 9.2 Responses:

- 1935 • 200: OK
- 1936 • 403: Forbidden
- 1937 • 404: Not found

1938 9.3 Example - retrieving server properties:

```
1939 >>Request
1940
1941 GET / HTTP/1.1
1942 Host: example.com
1943 ACCEPT:application/xrd+xml
1944
1945 >>Response
1946 <XRD xmlns="http://docs.oasis-open.org/ns/xri/xrd-1.0"
1947     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
1948   <Expires>1970-01-01T00:00:00Z</Expires>
1949   <Subject>http://example.com/calws</Subject>
1950   <Property type="http://docs.oasis-open.org/ns/wscal/calws/created"
1951     >1970-01-01</Property>
1952
1953   <Link rel="http://docs.oasis-open.org/ns/wscal/calws/timezone-service"
1954     href="http://example.com/tz" />
1955
1956   <calWS:privilege-set>
1957   <calWS:privilege><calWS:read></calWS:privilege>
1958   </calWS:privilege-set>
1959
1960   <Link rel="http://docs.oasis-open.org/ns/wscal/calws/principal-home"
1961     type="collection"
1962     href="http://example.com/calws/user/fred">
1963   <Title xml:lang="en">Fred's calendar home</Title>
1964   </Link>
1965
1966   <Link rel="http://docs.oasis-open.org/ns/wscal/calws/child-collection"
1967     type="calendar,scheduling"
1968     href="http://example.com/calws/user/fred/calendar">
1969   <Title xml:lang="en">Calendar</Title>
```

1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980

```
</Link>
  <Property type="http://docs.oasis-open.org/ns/wscal/calws/max-instances"
    >1000</Property>
  <Property type="http://docs.oasis-open.org/ns/wscal/calws/max-attendees-
per-instance"
    >100</Property>
</XRD>
```

1981
1982
1983
1984

1985
1986

1987
1988
1989

1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021

10 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.

10.1 Request parameters

- action=create

10.2 Responses:

- 201: created
- 403: Forbidden - no access

10.3 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;

- 2022 • **CalWS:after-max-date-time:** The resource submitted in the PUT request, or targeted by a COPY
2023 or MOVE request, MUST have all of its iCalendar DATE or DATE-TIME property values (for each
2024 recurring instance) less than the value of the CalDAV:max-date-time property value on the calendar
2025 collection where the resource will be stored;
- 2026 • **CalWS:too-many-instances:** The resource submitted in the PUT request, or targeted by a COPY
2027 or MOVE request, MUST generate a number of recurring instances less than or equal to the value
2028 of the CalDAV: max-instances property value on the calendar collection where the resource will be
2029 stored;
- 2030 • **CalWS:too-many-attendees-per-instance:** The resource submitted in the PUT request, or
2031 targeted by a COPY or MOVE request, MUST have a number of ATTENDEE properties on any one
2032 instance less than or equal to the value of the CalDAV:max-attendees-per-instance property value
2033 on the calendar collection where the resource will be stored;

2034 10.4 Example - successful POST:

```

2035 >>Request
2036
2037 POST /user/fred/calendar/?action=create HTTP/1.1
2038 Host: example.com
2039 Content-Type: application/xml+calendar; charset="utf-8"
2040 Content-Length: ?
2041
2042 <?xml version="1.0" encoding="utf-8" ?>
2043 <icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
2044   <vcalendar>
2045     ...
2046   </vcalendar>
2047 </icalendar>
2048
2049 >>Response
2050
2051 HTTP/1.1 201 Created
2052 Location: http://example.com/user/fred/calendar/event1.ics

```

2053 10.5 Example - unsuccessful POST:

```

2054 >>Request
2055
2056 POST /user/fred/readcalendar/?action=create HTTP/1.1
2057 Host: example.com
2058 Content-Type: text/text; charset="utf-8"
2059 Content-Length: ?
2060
2061 This is not an xml calendar object
2062
2063 >>Response
2064
2065 HTTP/1.1 403 Forbidden
2066 <?xml version="1.0" encoding="utf-8"
2067   xmlns:D="DAV:"
2068   xmlns:C="urn:ietf:params:xml:ns:caldav" ?>
2069 <D:error>
2070   <C:supported-calendar-data/>
2071   <D:description>Not an icalendar object</C:description>
2072 </D:error>

```

2073

11 Retrieving resources

2074 A simple GET on the href will return a named resource. If that resource is a recurring event or task with
2075 overrides, the entire set will be returned. The desired format is specified in the ACCEPT header. The
2076 default form is application/xml+calendar

11.1 Request parameters

- none

11.2 Responses:

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

11.3 Example - successful fetch:

```
2084 >>Request
2085
2086 GET /user/fred/calendar/event1.ics HTTP/1.1
2087 Host: example.com
2088
2089 >>Response
2090
2091 HTTP/1.1 200 OK
2092 Content-Type: application/xml+calendar; charset="utf-8"
2093 Content-Length: ?
2094
2095 <?xml version="1.0" encoding="utf-8" ?>
2096 <icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
2097   <vcalendar>
2098     ...
2099   </vcalendar>
2100 </icalendar>
```

11.4 Example - unsuccessful fetch:

```
2102 >>Request
2103
2104 PUT /user/fred/calendar/noevent1.ics HTTP/1.1
2105 Host: example.com
2106
2107 >>Response
2108
2109 HTTP/1.1 404 Not found
```

2110

12 Updating resources

2111 Resources are updated with the PUT method targeted at the resource href. The body of the request
2112 contains a complete new resource which effectively replaces the targeted resource. To allow for optimistic
2113 locking of the resource use the if-match header.

2114 When updating a recurring event all overrides and master must be supplied as part of the content.

2115 Preconditions as specified in Section 10.3 are applicable.

2116 12.1 Responses:

- 2117 • 200: OK
- 2118 • 304: Not modified - entity was modified by some other request
- 2119 • 403: Forbidden - no access, does not exist etc. See error response

2120

2121

Example 16: Successful Update

```
2122 >>Request
2123
2124 PUT /user/fred/calendar/event1.ics HTTP/1.1
2125 Host: example.com
2126 Content-Type: application/xml+calendar; charset="utf-8"
2127 Content-Length: ?
2128
2129 <?xml version="1.0" encoding="utf-8" ?>
2130 <icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
2131   <vcalendar>
2132     ...
2133   </vcalendar>
2134 </icalendar>
2135
2136 >>Response
2137
2138 HTTP/1.1 200 OK
```

2139

Example 17: Unsuccessful Update

```
2140 >>Request
2141
2142 PUT /user/fred/readcalendar/event1.ics HTTP/1.1
2143 Host: example.com
2144 Content-Type: application/xml+calendar; charset="utf-8"
2145 Content-Length: ?
2146
2147 <?xml version="1.0" encoding="utf-8" ?>
2148 <icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
2149   <vcalendar>
2150     ...
2151   </vcalendar>
2152 </icalendar>
2153
2154 >>Response
2155
2156 HTTP/1.1 403 Forbidden
2157 Content-Type: application/xml; charset="utf-8"
2158 Content-Length: xxxx
2159
2160 <?xml version="1.0" encoding="utf-8"
```

```
2161     xmlns:D="DAV:"
2162     xmlns:CW=" http://docs.oasis-open.org/ws-calendar/CalWS" ?>
2163 <CW:error>
2164   <CW:target-exists/>
2165   <CW:description>Target of update must exist</C:description>
2166 </CW:error>
```


2167

13 Deletion of resources

2168 Delete is defined in [RFC 2616] Section 9.7. In addition to conditions defined in that specification, servers
2169 must remove any references from the deleted resource to other resources. Resources are deleted with
2170 the DELETE method targeted at the resource URL. After a successful completion of a deletion a GET on
2171 that URL must result in a 404 - Not Found status.

2172

13.1 Delete for Collections

2173 Delete for collections may or may not be supported by the server. Certain collections are considered
2174 undeletable. On a successful deletion of a collection all contained resources to any depth must also be
2175 deleted.

2176

13.2 Responses:

2177

- 200: OK

2178

- 403: Forbidden - no access

2179

- 404: Not Found

2180 14 Querying calendar resources

2181 Querying provides a mechanism by which information can be obtained from the service through possibly
2182 complex queries. A list of icalendar properties can be specified to limit the amount of information returned
2183 to the client. A query takes the parts

- 2184 • Limitations on the data returned
- 2185 • Selection of the data
- 2186 • Optional timezone id for floating time calculations.

2187 The current specification uses CalDAV multiget and calendar-query XML bodies as specified in [RFC
2188 4791] with certain limitations and differences.

- 2189 1. The POST method is used for all requests, the action being identified by the outer element.
- 2190 2. While CalDAV servers generally only support [RFC 5545] and assume that as the default, the
2191 delivery format for CalWS will, by default, be [draft-xcal].
- 2192 3. The CalDAV query allows the specification of a number of DAV properties. Specification of these
2193 properties, with the exception of DAV:getetag, is considered an error in CalWS.
- 2194 4. The CalDAV:propnames element is invalid

2195 With those differences, the CalDAV specification is the normative reference for this operation.

2196 14.1 Limiting data returned

2197 This is achieved by specifying one of the following

- 2198 • CalDAV:allprop return all properties (some properties are specified as not being part of the allprop
2199 set so are not returned)
- 2200 • CalDAV:prop An element which contains a list of properties to be returned . May only contain
2201 DAV:getetag and CalDAV:calendar-data

2202 Of particular interest, and complexity, is the calendar-data property which can contain a time range to limit
2203 the range of recurrences returned and/or a list of calendar properties to return.

2204 14.2 Pre/postconditions for calendar queries

2205 The preconditions as defined in in [RFC 4791] Section 7.8 apply here. CalDav errors may be reported by
2206 the service when preconditions or postconditions are violated.

2207 14.3 Example: time range limited retrieval

2208 This example shows the time-range limited retrieval from a calendar which results in 2 events, one a
2209 recurring event and one a simple non-recurring event.

```
2210 >> Request <<
2211
2212 POST /user/fred/calendar/ HTTP/1.1
2213 Host: calws.example.com
2214 Depth: 1
2215 Content-Type: application/xml; charset="utf-8"
2216 Content-Length: xxxx
2217
2218 <?xml version="1.0" encoding="utf-8" ?>
2219 <C:calendar-query xmlns:D="DAV:"
2220                  xmlns:C="urn:ietf:params:xml:ns:caldav">
2221   <D:prop>
2222     <D:getetag/>
2223     <C:calendar-data content-type="application/xml+calendar" >
```

```

2224     <C:comp name="VCALENDAR">
2225         <C:prop name="VERSION"/>
2226         <C:comp name="VEVENT">
2227             <C:prop name="SUMMARY"/>
2228             <C:prop name="UID"/>
2229             <C:prop name="DTSTART"/>
2230             <C:prop name="DTEND"/>
2231             <C:prop name="DURATION"/>
2232             <C:prop name="RRULE"/>
2233             <C:prop name="RDATE"/>
2234             <C:prop name="EXRULE"/>
2235             <C:prop name="EXDATE"/>
2236             <C:prop name="RECURRENCE-ID"/>
2237         </C:comp>
2238     </C:comp>
2239 </C:calendar-data>
2240 </D:prop>
2241 <C:filter>
2242     <C:comp-filter name="VCALENDAR">
2243         <C:comp-filter name="VEVENT">
2244             <C:time-range start="20060104T000000Z"
2245                 end="20060105T000000Z"/>
2246         </C:comp-filter>
2247     </C:comp-filter>
2248 </C:filter>
2249 </C:calendar-query>
2250
2251 >> Response <<
2252
2253 HTTP/1.1 207 Multi-Status
2254 Date: Sat, 11 Nov 2006 09:32:12 GMT
2255 Content-Type: application/xml; charset="utf-8"
2256 Content-Length: xxxx
2257
2258 <?xml version="1.0" encoding="utf-8" ?>
2259 <D:multistatus xmlns:D="DAV:"
2260     xmlns:C="urn:ietf:params:xml:ns:caldav">
2261     <D:response>
2262         <D:href>http://cal.example.com/bernard/work/abcd2.ics</D:href>
2263         <D:propstat>
2264             <D:prop>
2265                 <D:getetag>"fffff-abcd2"</D:getetag>
2266                 <C:calendar-data content-type="application/xml+calendar" >
2267                     <xc:icalendar
2268                         xmlns:xc="urn:ietf:params:xml:ns:icalendar-2.0">
2269                 <xc:vcalendar>
2270                     <xc:properties>
2271                         <xc:calscale><text>GREGORIAN</text></xc:calscale>
2272                         <xc:prodid>
2273                             <xc:text>-//Example Inc.//Example Calendar//EN</xc:text>
2274                         </xc:prodid>
2275                         <xc:version><xc:text>2.0</xc:text></xc:version>
2276                     </xc:properties>
2277                     <xc:components>
2278                         <xc:vevent>
2279                             <xc:properties>
2280                                 <xc:dtstart>
2281                                     <xc:parameters>
2282                                         <xc:tzid>US/Eastern<xc:tzid>
2283                                     <xc:parameters>
2284                                         <xc:date-time>20060102T120000</xc:date-time>
2285                                 </xc:dtstart>
2286                                 <xc:duration><xc:duration>PT1H</xc:duration></xc:duration>
2287                             <xc:summary>

```

```

2288     <xc:text>Event #2</xc:text>
2289 </xc:summary>
2290 <xc:uid>
2291     <xc:text>00959BC664CA650E933C892C@example.com</xc:text>
2292 </xc:uid>
2293 <xc:rrule>
2294     <xc:recur>
2295         <xc:freq>DAILY</xc:freq>
2296         <xc:count>5</xc:count>
2297     </xc:recur>
2298 </xc:rrule>
2299 </xc:properties>
2300 </xc:vevent>
2301
2302 <xc:vevent>
2303     <xc:properties>
2304         <xc:dtstart>
2305             <xc:parameters>
2306                 <xc:tzid>US/Eastern<xc:tzid>
2307             <xc:parameters>
2308                 <xc:date-time>20060104T140000</xc:date-time>
2309             </xc:dtstart>
2310         <xc:duration><xc:duration>PT1H</xc:duration></xc:duration>
2311         <xc:summary>
2312             <xc:text>Event #2 bis</xc:text>
2313         </xc:summary>
2314         <xc:uid>
2315             <xc:text>00959BC664CA650E933C892C@example.com</xc:text>
2316         </xc:uid>
2317         <xc:recurrence-id>
2318             <xc:parameters>
2319                 <xc:tzid>US/Eastern<xc:tzid>
2320             <xc:parameters>
2321                 <xc:date-time>20060104T120000</xc:date-time>
2322             </xc:recurrence-id>
2323         <xc:rrule>
2324             <xc:recur>
2325                 <xc:freq>DAILY</xc:freq>
2326                 <xc:count>5</xc:count>
2327             </xc:recur>
2328         </xc:rrule>
2329     </xc:properties>
2330 </xc:vevent>
2331
2332 <xc:vevent>
2333     <xc:properties>
2334         <xc:dtstart>
2335             <xc:parameters>
2336                 <xc:tzid>US/Eastern<xc:tzid>
2337             <xc:parameters>
2338                 <xc:date-time>20060106T140000</xc:date-time>
2339             </xc:dtstart>
2340         <xc:duration><xc:duration>PT1H</xc:duration></xc:duration>
2341         <xc:summary>
2342             <xc:text>Event #2 bis bis</xc:text>
2343         </xc:summary>
2344         <xc:uid>
2345             <xc:text>00959BC664CA650E933C892C@example.com</xc:text>
2346         </xc:uid>
2347         <xc:recurrence-id>
2348             <xc:parameters>
2349                 <xc:tzid>US/Eastern<xc:tzid>
2350             <xc:parameters>
2351                 <xc:date-time>20060106T120000</xc:date-time>

```

```

2352     </xc:recurrence-id>
2353     <xc:rrule>
2354         <xc:recur>
2355             <xc:freq>DAILY</xc:freq>
2356             <xc:count>5</xc:count>
2357         </xc:recur>
2358     </xc:rrule>
2359 </xc:properties>
2360 </xc:vevent>
2361 </xc:components>
2362 </xc:vcalendar>
2363 </xc:icalendar>
2364     </C:calendar-data>
2365 </D:prop>
2366     <D:status>HTTP/1.1 200 OK</D:status>
2367 </D:propstat>
2368 </D:response>
2369 <D:response>
2370     <D:href>http://cal.example.com/bernard/work/abcd3.ics</D:href>
2371     <D:propstat>
2372         <D:prop>
2373             <D:getetag>"fffff-abcd3"</D:getetag>
2374             <C:calendar-data content-type="application/xml+calendar" >
2375                 <xcal:icalendar
2376                     xmlns:xc="urn:ietf:params:xml:ns:icalendar-2.0">
2377 <xc:vcalendar>
2378 <xc:properties>
2379 <xc:calscale><text>GREGORIAN</text></xc:calscale>
2380 <xc:prodid>
2381 <xc:text>-//Example Inc.//Example Calendar//EN</xc:text>
2382 </xc:prodid>
2383 <xc:version><xc:text>2.0</xc:text></xc:version>
2384 </xc:properties>
2385 <xc:components>
2386 <xc:vevent>
2387 <xc:properties>
2388 <xc:dtstart>
2389 <xc:parameters>
2390 <xc:tzid>US/Eastern<xc:tzid>
2391 <xc:parameters>
2392 <xc:date-time>20060104T100000</xc:date-time>
2393 </xc:dtstart>
2394 <xc:duration><xc:duration>PT1H</xc:duration></xc:duration>
2395 <xc:summary>
2396 <xc:text>Event #3</xc:text>
2397 </xc:summary>
2398 <xc:uid>
2399 <xc:text>DC6C50A017428C5216A2F1CD@example.com</xc:text>
2400 </xc:uid>
2401 <xc:rrule>
2402 <xc:recur>
2403 <xc:freq>DAILY</xc:freq>
2404 <xc:count>5</xc:count>
2405 </xc:recur>
2406 </xc:rrule>
2407 </xc:properties>
2408 </xc:vevent>
2409 </xc:components>
2410 </xc:vcalendar>
2411 </xc:icalendar>
2412     </C:calendar-data>
2413 </D:prop>
2414     <D:status>HTTP/1.1 200 OK</D:status>
2415 </D:propstat>

```

2416
2417

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

2418

15 Free-busy queries

2419 Free-busy queries are used to obtain free-busy information for a calendar-collection or principals. The
2420 result contains information only for events to which the current principal has sufficient access.

2421 When targeted at a calendar collection the result is based only on the calendaring entities contained in
2422 that collection. When targeted at a principal free-busy URL the result will be based on all information
2423 which affect the principals free-busy status, for example availability.

2424 The possible targets are:

- 2425 • A calendar collection URL
- 2426 • The XRD link with relation CalWS/current-principal-freebusy
- 2427 • The XRD link with relation CalWS/principal-freebusy with a principal given in the request.

2428 The query follows the specification defined in Error! Reference source not found. with certain limitations.
2429 As an authenticated user to the CalWS service scheduling read-freebusy privileges must have been
2430 granted. As an unauthenticated user equivalent access must have been granted to unauthenticated
2431 access.

2432 Freebusy information is returned by default as xcalendar vfreebusy components, as defined by **[draft-
2433 xcal]**. Such a component is not meant to conform to the requirements of VFREEBUSY components in
2434 **[RFC 5546]**. The VFREEBUSY component SHOULD conform to section "4.6.4 Free/Busy Component" of
2435 **[RFC 5545]**. A client SHOULD ignore the ORGANIZER field..

2436 Since a Freebusy query can only refer to a single user, a client will already know how to match the result
2437 component to a user. A server MUST only return a single vfreebusy component.

2438 15.1 ACCEPT header

2439 The Accept header is used to specify the format for the returned data. In the absence of a header the
2440 data should be returned as specified in **[draft-xcal]**, that is, as if the following had been specified

2441 `ACCEPT: application/xml+calendar`

2442 15.2 URL Query Parameters

2443 None of these parameters are required except for the conditions noted below. Appropriate defaults will be
2444 supplied by the server.

2445 15.2.1 start

2446 **Default:** The default value is left up to the server. It may be the current day, start of the current
2447 month, etc.

2448 **Description:** Specifies the start date for the Freebusy data. The server is free to ignore this value and
2449 return data in any time range. The client must check the data for the returned time range.

2450 **Format:** A profile of an **[RFC3339]** Date/Time. Fractional time is not supported. The server MUST
2451 support the expanded version e.g.

2452 `2007-01-02T13:00:00-08:00`

2453 It is up to the server to interpret local date/times.

2454 **Example:**

2455 `2007-02-03T15:30:00-0800`

2456 `2007-12-01T10:15:00Z`

2457 **Notes:** Specifying only a start date/time without specifying an end-date/time or period should be
2458 interpreted as in **[RFC 5545]**. The effective period should cover the remainder of that day.

2459 Date-only values are disallowed as the server cannot determine the correct start of the day. Only

2460 UTC or date/time with offset values are permitted.

2461 15.2.2 end

2462 **Default:** Same as start

2463 **Description:** Specifies the end date for the Freebusy data. The server is free to ignore this value.

2464 **Format:** Same as start

2465 **Example:** Same as start

2466 15.2.3 period

2467 **Default:** The default value is left up to the server. The recommended value is "P42D".

2468 **Description:** Specifies the amount of Freebusy data to return. A client cannot specify both a period
2469 and an end date. Period is relative to the start parameter.

2470 **Format:** A duration as defined in section 4.3.6 of [RFC 5545]

2471 **Example:**

2472 P42D

2473 15.2.4 account

2474 **Default:** none

2475 **Description:** Specifies the principal when the request is targeted at the XRD CalWS/principal-
2476 freebusy. Specification of this parameter is an error otherwise.

2477 **Format:** Server specific

2478 **Example:**

2479 fred
2480 /principals/users/jim
2481 user1@example.com

2482 15.3 URL parameters - notes

2483 The server is free to ignore the start, end and period parameters. It is recommended that the server return
2484 at least 6 weeks of data from the current day.

2485 A client MUST check the time range in the VFREEBUSY response as a server may return a different time
2486 range than the requested range.

2487 15.4 HTTP Operations

2488 The server SHOULD return an Etag response header for a successful GET request targeting a Freebusy
2489 read URL. Clients MAY use the Etag response header value to do subsequent "conditional" GET
2490 requests that will avoid re-sending the Freebusy data again if it has not changed.

2491 15.5 Response Codes

2492 Below are the typical status codes returned by a GET request targeting a Free-busy URL. Note that other
2493 HTTP status codes not listed here might also be returned by a server.

- 2494
- 200 OK
 - 302 Found
 - 400 Start parameter could not be understood / End parameter could not be understood / Period
2497 parameter could not be understood
 - 401 Unauthorized
- 2498

- 2499 • 403 Forbidden
- 2500 • 404 The data for the requested principal is not currently available, but may be available later.
- 2501 • 406 The requested format in the accept header is not supported.
- 2502 • 410 The data for the requested principal is no longer available
- 2503 • 500 General server error

2504 15.6 Examples

2505 The following are examples of URLs used to retrieve Free-busy data for a user:

```
2506 http://www.example.com/freebusy/user1@example.com?
2507 start=2007-09-01T00:00:00-08:00&end=2007-09-31T00:00:00-08:00
2508
2509 http://www.example.com/freebusy/user1@example.com?
2510 start=2007-09-01T00:00:00-08:00&end=2007-09-31T00:00:00-08:00
2511
2512 http://www.example.com/freebusy/user1@example.com
2513
2514 http://www.example.com/freebusy?user=user%201@example.com&
2515 start=2008-01-01T00:00:00Z&end=2008-12-31T00:00:00Z
```

2516 Some Request/Response Examples:

2517 A URL with no query parameters:

```
2518 >> Request <<
2519 GET /freebusy/bernard/ HTTP/1.1
2520 Host: www.example.com
2521
2522 >> Response <<
2523 HTTP/1.1 200 OK
2524 Content-Type: application/xml+calendar; charset="utf-8"
2525 Content-Length: xxxx
2526
2527 <xc:icalendar xmlns:xc="urn:ietf:params:xml:ns:icalendar-2.0">
2528   <xc:vcalendar>
2529     <xc:properties>
2530       <xc:calscale><text>GREGORIAN</text></xc:calscale>
2531       <xc:prodid>
2532         <xc:text>-//Example Inc.//Example Calendar//EN</xc:text>
2533       </xc:prodid>
2534       <xc:version><xc:text>2.0</xc:text></xc:version>
2535     </xc:properties>
2536     <xc:components>
2537       <xc:vfreebusy>
2538         <xc:properties>
2539           <xc:uid>
2540             <xc:text>76ef34-54a3d2@example.com</xc:text>
2541           </xc:uid>
2542           <xc:dtstart>
2543             <xc:date-time>20060101T000000Z</xc:date-time>
2544           </xc:dtstart>
2545           <xc:dtend>
2546             <xc:date-time>20060108T000000Z</xc:date-time>
2547           </xc:dtend>
2548           <xc:dtstamp>
2549             <xc:date-time>20050530T123421Z</xc:date-time>
2550           </xc:dtstamp>
2551           <xc:freebusy>
2552             <xc:parameters>
2553               <xc:fbtype>BUSYTENTATIVE<xc:fbtype>
2554             <xc:parameters>
2555               <xc:period>20060102T100000Z/20060102T120000Z</xc:period>
```

```

2556     </xc:freebusy>
2557     <xc:freebusy>
2558         <xc:period>20060103T100000Z/20060103T120000Z</xc:period>
2559     </xc:freebusy>
2560     <xc:freebusy>
2561         <xc:period>20060104T100000Z/20060104T120000Z</xc:period>
2562     </xc:freebusy>
2563     <xc:freebusy>
2564         <xc:parameters>
2565             <xc:fbtype>BUSYUNAVAILABLE<xc:fbtype>
2566         <xc:parameters>
2567         <xc:period>20060105T100000Z/20060105T120000Z</xc:period>
2568     </xc:freebusy>
2569     <xc:freebusy>
2570         <xc:period>20060106T100000Z/20060106T120000Z</xc:period>
2571     </xc:freebusy>
2572 </xc:vfreebusy>
2573 </xc:components>
2574 </xc:vcalendar>
2575 <xc:icalendar>

```

2576 A URL with start and end parameters:

```

2577 >> Request <<
2578 GET /freebusy/user1@example.com?start=2007-09-01T00:00:00-08:00&end=2007-09-
2579 31T00:00:00-08:00
2580 HTTP/1.1
2581 Host: www.example.com
2582
2583 >> Response <<
2584 HTTP/1.1 200 OK
2585 Content-Type: application/xml+calendar; charset="utf-8"
2586 Content-Length: xxxx
2587
2588 <xc:icalendar xmlns:xc="urn:ietf:params:xml:ns:icalendar-2.0">
2589     <xc:vcalendar>
2590         <xc:properties>
2591             <xc:calscale><text>GREGORIAN</text></xc:calscale>
2592             <xc:prodid>
2593                 <xc:text>-//Example Inc.//Example Calendar//EN</xc:text>
2594             </xc:prodid>
2595             <xc:version><xc:text>2.0</xc:text></xc:version>
2596         </xc:properties>
2597         <xc:components>
2598             <xc:vfreebusy>
2599                 <xc:properties>
2600                     <xc:uid>
2601                         <xc:text>76ef34-54a3d2@example.com</xc:text>
2602                     </xc:uid>
2603                     <xc:dtstart>
2604                         <xc:date-time>20070901T000000Z</xc:date-time>
2605                     </xc:dtstart>
2606                     <xc:dtend>
2607                         <xc:date-time>20070931T000000Z</xc:date-time>
2608                     </xc:dtend>
2609                     <xc:dtstamp>
2610                         <xc:date-time>20050530T123421Z</xc:date-time>
2611                     </xc:dtstamp>
2612                     <xc:freebusy>
2613                         <xc:period>20070915T230000Z/20070916T010000Z</xc:period>
2614                     </xc:freebusy>
2615                 </xc:vfreebusy>
2616             </xc:components>
2617         </xc:vcalendar>
2618     </xc:icalendar>

```

2619 A URL for which the server does not have any data for that user:

```
2620 >> Request <<
2621 GET /freebusy/user1@example.com?start=2012-12-01T00:00:00-08:00&end=2012-12-
2622 31T00:00:00-08:00
2623 HTTP/1.1
2624 Host: www.example.com
2625
2626 >> Response <<
2627 HTTP/1.1 404 No data
2628
```

2629 **16 Conformance**

2630 WS-Calendar Intervals SHALL have a Duration. Intervals MAY have a StartTime. Intervals SHALL NOT
2631 include an END time. If a non-compliant Interval is received with an END time, it may be ignored.

2632 A performance component SHALL not include Start, Stop, and Duration elements. Two out of the three
2633 elements is acceptable, but not three.

2634 In Partitions, the Description, Summary and Priority of each Interval SHALL be excluded.

2635 An Calendar Gluon may either have a dtStart or a dtEnd, but may not have both.

2636 *All OASIS specifications require conformance*

2637

A. Acknowledgements

2638 The following individuals have participated in the creation of this specification and are gratefully
2639 acknowledged:

2640 **Participants:**

2641 Bruce Bartell, Southern California Edison
2642 Brad Benson, Trane
2643 Edward Cazalet, Individual
2644 Toby Considine, University of North Carolina at Chapel Hill
2645 William Cox, Individual
2646 Sharon Dinges, Trane
2647 Craig Gemmill, Tridium, Inc.
2648 Girish Ghatikar, Lawrence Berkeley National Laboratory
2649 Gerald Gray, Southern California Edison
2650 Gale Horst, Electric Power Research Institute (EPRI)
2651 Gershon Janssen, Individual
2652 Ed Koch, Akuacom Inc.
2653 Benoit Lepeuple, LonMark International*
2654 Carl Mattocks, CheckMi*
2655 Robert Old, Siemens AG
2656 Alexander Papaspyrou, Technische Universitat Dortmund
2657 Jeremy J. Roberts, LonMark International
2658 David Thewlis, CalConnect

2659

2660

2661 The Calendaring and Scheduling Consortium (CalConnect) TC-XML committee worked closely with WS-
2662 Calendar Technical Committee, bridging to developing IETF standards and contributing the Services
2663 definitions that make up Services beginning in Section 7, Calendar Services. The Technical Committee
2664 gratefully acknowledges their assistance and cooperation as well. Contributors to TC XML include:

2665 Cyrus Daboo, Apple
2666 Mike Douglas, Rensselaer Polytechnic Institute
2667 Steven Lees, Microsoft
2668 Tong Li, IBM

2669

2670

B. An Introduction to Internet Calendaring

2671 *The WS-Calendar Technical Committee thanks CalConnect for contributing this overview of iCalendar*
2672 *and its use.*

B.1 icalendar

B.1.1 History

2675 The iCalendar specification was first produced by the IETF in 1998 as RFC 2445 [1]. Since then it has
2676 become the dominant standard for calendar data interchange on the internet and between devices
2677 (desktop computers, mobile phones etc.). The specification was revised in 2009 as RFC 5545 [4].

2678 Alongside iCalendar is the iTIP specification (RFC 2446 [2] and revised as RFC 5546[5]) that defines how
2679 iCalendar is used to carry out scheduling operations (for example, how an organizer can invite attendees
2680 to a meeting and receive their replies). This forms the basis for email-based scheduling using iMIP (the
2681 specification that describes how to use iTIP with email - RFC 2447 [3]).

2682 iCalendar itself is a text-based data format. However, an XML format is also available, providing a one-to-
2683 one mapping to the text format (draft [7]).

2684 iCalendar data files typically have a .ics file name extension. Most desktop calendar clients can import or
2685 export iCalendar data, or directly access such data over the Internet using a variety of protocols.

B.1.2 Data model

2687 The iCalendar data format has a well defined data model. "iCalendar objects" encompass a set of
2688 "iCalendar components" each of which contains a set of "iCalendar properties" and possibly other sub-
2689 components. An iCalendar property consists of a name, a set of optional parameters (specified as "key-
2690 value" pairs) and a value.

2691 iCalendar components include:

2692 "VEVENT" which represents an event

2693 "VTODO" which represents a task or to-do

2694 "VJOURNAL" which represents a journal entry

2695 "VFREEBUSY" which represents periods of free or busy time information

2696 "VTIMEZONE" which represents a timezone definition (timezone offset and daylight saving rules)

2697 "VALARM" is currently the only defined sub-component and is used to set alarms or reminders on events
2698 or tasks.

2699 Properties include:

2700 "DTSTART" which represents a start time for a component

2701 "DTEND" which represents an end time for a component

2702 "SUMMARY" which represents a title or summary for a component

2703 "RRULE" which can specify rules for repeating events or tasks (for example, every day, every week on
2704 Tuesdays, etc.)

2705 "ORGANIZER" which represents the calendar user who is organizing an event or assigning a task

2706 "ATTENDEE" which represents calendar users attending an event or assigned a task

2707 In addition to this data model and the pre-defined properties, the specification defines how all those are
2708 used together to define the semantics of calendar objects and scheduling. The semantics are basically a
2709 set of rules stating how all the components and properties are used together to ensure that all iCalendar
2710 products can work together to achieve good interoperability. For example, a rule requires that all events
2711 must have one and only one "DTSTART" property. The most important part of the iCalendar specification

2712 is the semantics of the calendaring model that it represents. The use of text or XML to encode those is
2713 secondary.

2714 **B.1.3 Scheduling**

2715 The iTIP specification defines how iCalendar objects are exchanged in order to accomplish the key task
2716 needed to schedule events or tasks. An example of a simple workflow is as follows:

- 2717 1. To schedule an event, an organizer creates the iCalendar object representing the event and adds
2718 calendar users as attendees.
- 2719 2. The organizer then sends an iTIP "REQUEST" message to all the attendees.
- 2720 3. Upon receipt of the scheduling message, each attendee can decide whether they want to attend
2721 the meeting or not.
- 2722 4. Each attendee can then respond back to the organizer using an iTIP "REPLY" message
2723 indicating their own attendance status.

2724 iTIP supports other types of scheduling messages, for example, to cancel meetings, add new instances to
2725 a repeating meeting, etc.

2726 **B.1.4 Extensibility**

2727 iCalendar was designed to be extensible, allowing for new components, properties and parameters to be
2728 defined as needed. A registry exists to maintain the list of standard extensions with references to their
2729 definitions to ensure anyone can use them and work well with others.

2730 **B.2 Calendar data access and exchange protocols**

2731 **B.2.1 Internet Calendar Subscriptions**

2732 An Internet calendar subscription is simply an iCalendar data file made available on a web server. Users
2733 can use this data in two ways:

- 2734 – The data can be downloaded from the web server and then imported directly into an iCalendar
2735 aware client. This solution works well for calendar data that is not likely to change over time (for
2736 example the list of national holidays for the next year).
- 2737 – Calendar clients that support "direct" subscriptions can use the URL to the calendar data on the
2738 web server to download the calendar data themselves. Additionally, the clients can check the web
2739 server on a regular basis for updates to the calendar data, and then update their own cached
2740 copy of it. This allows calendar data that changes over time to be kept synchronized.

2741 **B.2.2 CalDAV**

2742 CalDAV is a calendar access protocol and is defined in RFC 4791 [6]. The protocol is based on WebDAV
2743 which is an extension to HTTP that provides enhanced capabilities for document management on web
2744 servers.

2745 CalDAV is used in a variety of different environments, ranging from very large internet service providers,
2746 to large and small corporations or institutions, and to small businesses and individuals.

2747 CalDAV clients include desktop applications, mobile devices and browser-based solutions. It can also be
2748 used by "applets", for example, a web page panel that displays a user's upcoming events.

2749 One of the key aspects of CalDAV is its data model. Simply put, it defines a "calendar home" for each
2750 calendar user, within which any number of "calendars" can be created. Each "calendar" can contain any
2751 number of iCalendar objects representing individual events, tasks or journal entries. This data model
2752 ensures that clients and servers can interoperate well.

2753 In addition to providing simple operations to read, write and delete calendar data, CalDAV provides a
2754 querying mechanism to allow clients to fetch calendar data matching specific criteria. This is commonly

2755 used by clients to do "time-range" queries, i.e., find the set of events that occur within a given start/end
2756 time period.

2757 CalDAV also supports access control allowing for features such as delegated calendars and calendar
2758 sharing.

2759 CalDAV also specifies how scheduling operations can be done using the protocol. Whilst it uses the
2760 semantics of the iTIP protocol, it simplifies the process by allowing simple calendar data write operations
2761 to trigger the sending of scheduling messages, and it has the server automatically process the receipt of
2762 scheduling messages. Scheduling can be done with other users on the CalDAV server or with calendar
2763 users on other systems (via some form of "gateway").

2764 **B.2.3 ActiveSync/SyncML**

2765 ActiveSync and SyncML are technologies that allow multiple devices to synchronize data with a server,
2766 with calendar data being one of the classes of data supported. These have typically been used for low-
2767 end and high-end mobile devices.

2768 **B.2.4 CalWS**

2769 CalWS is a web services calendar access API developed by The Calendaring and Scheduling
2770 Consortium and the OASIS organization, to be used as part of the Oasis WS-Calendar standard. It
2771 provides an API to access and manipulate calendar data stored on a server. It follows a similar data
2772 model to CalDAV and has been designed to co-exist with a CalDAV service offering the same data.

2773 **B.2.5 iSchedule**

2774 iSchedule is a protocol to allow scheduling between users on different calendaring systems and across
2775 different internet domains. It transports iTIP scheduling messages using HTTP between servers. Servers
2776 use DNS and various security mechanisms to determine the authenticity of messages received.

2777 It has been specifically designed to be independent of any calendar system in use at the endpoints, so
2778 that it is compatible with many different systems. This allows organizations with different calendar
2779 systems to exchange scheduling messages with each other, and also allows a single organization with
2780 multiple calendar systems (for example due to mergers, or different departmental requirements) to
2781 exchange scheduling messages between users of each system.

2782 **B.3 References**

2783 [1] <https://datatracker.ietf.org/doc/rfc2445/> : 'Internet Calendaring and Scheduling Core Object
2784 Specification'

2785 [2] <https://datatracker.ietf.org/doc/rfc2446/> : 'iCalendar Transport-Independent Interoperability Protocol'

2786 [3] <https://datatracker.ietf.org/doc/rfc2447/> : 'iCalendar Message-Based Interoperability Protocol'

2787 [4] <https://datatracker.ietf.org/doc/rfc5545/> : 'Internet Calendaring and Scheduling Core Object
2788 Specification'

2789 [5] <https://datatracker.ietf.org/doc/rfc5546/> : 'iCalendar Transport-Independent Interoperability Protocol'

2790 [6] <https://datatracker.ietf.org/doc/rfc4791/> : 'Calendaring Extensions to WebDAV'

2791 [7] <https://datatracker.ietf.org/doc/draft-daboo-et-al-icalendar-in-xml/> : 'xCal: The XML format for
2792 iCalendar'

2793

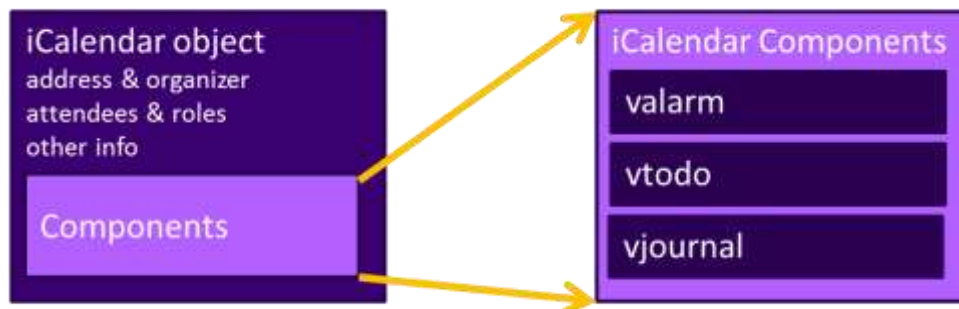
2794
2795

C. Overview of WS-Calendar, its Antecedents and its Use

2796 iCalendar has long been the predominant message format for an Internet user to send meeting requests
2797 and tasks to other Internet users by email. The recipient can respond to the sender easily or counter
2798 propose another meeting date/time. iCalendar support is built into all major email systems and email
2799 clients. While SMTP is the predominant means to transport iCalendar messages, protocols including
2800 WebDAV and SyncML are used to transport collections of iCalendar information. No similar standard for
2801 service interactions has achieved similar widespread use.

2802 The Calendar and Scheduling Consortium (CalConnect), working within the IETF, updated the iCalendar
2803 standard in the summer of 2009 to support extension ([RFC5545]). In 2010, the same group defined
2804 [XCAL], a canonical XML serialization for iCalendar, currently (08/21/2008) on the recommended
2805 standards track within the IETF. This specification supports extensions, including handling non-standard,
2806 i.e., non-iCalendar, data during message storage and retrieval.

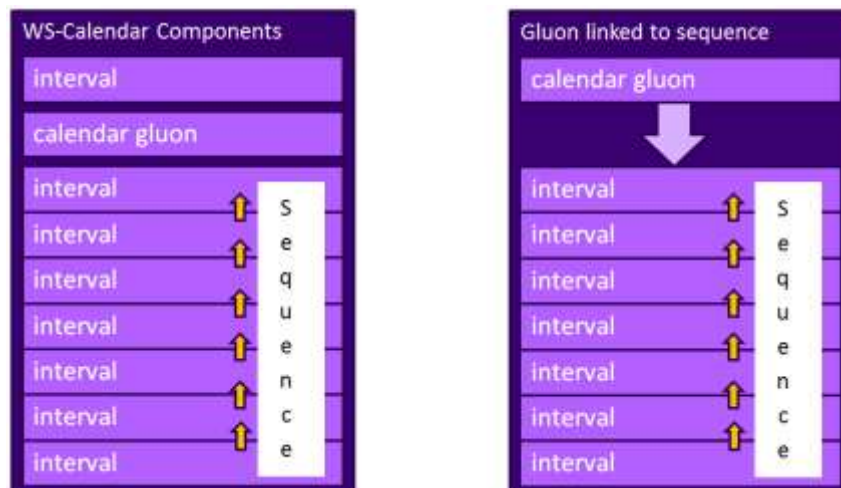
2807 WS-Calendar builds on this work, and consists of extensions to the vocabulary of iCalendar, along with
2808 standard services to extend calendaring and scheduling into service interactions. iCalendar consists of a
2809 number of fields that support the delivery, update, and synchronization of if calendar messages and a list
2810 of components. The components can specify defined relationships between each other.



2811
2812

Figure 3: iCalendar overview

2813 WS-Calendar defines the Interval, a profile of the vtodo component requiring only a duration and an
2814 artifact to define service delivery and performance. WS-Calendar also defines the CalendarGluon
2815 component, a container for holding only a service delivery and performance artifact, to associate with a
2816 component or group of components.



2817

2818

Figure 4: WS-Calendar and EMIX

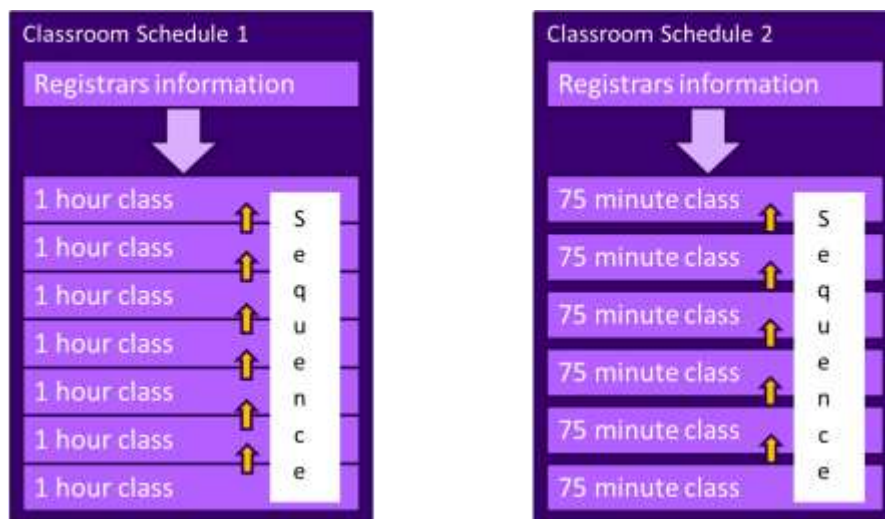
2819 A set of intervals that have defined temporal relationships is a Sequence. Temporal relationships express
2820 how the occurrence of one interval is related to another. For example, Interval B may begin 10 minutes
2821 after Interval A completes, or Interval D may start 5 minutes after Interval C starts. An Calendar Gluon
2822 linked to a Sequence defines service performance for all Intervals in the Sequence. Because each
2823 interval has its own service performance contract, specifications built on WS-Calendar can define rules for
2824 inheritance and over-rides with a sequence.

2825 The Partition is a sub-class of a Sequence in which all Intervals follow consecutively with no lag time.
2826 Intervals in a Partition normally have the same Duration, but WS-Calendar does support overriding the
2827 duration on an individual basis.

2828 C.1 Scheduling Sequences

2829 A Sequence is a general pattern of behaviors and results that does not require a specific schedule. A
2830 publishing service may advertise a Sequence with no schedule, i.e., no specific time for performance.
2831 When the Sequence is invoked or contracted, a specific performance time is added. In the original
2832 iCalendar components, this would add the starting date and time (dtStart) to the component. In WS-
2833 Calendar, we add the starting date and time only to the first Interval of a Sequence; the performance
2834 times for all other Intervals in the Sequence are derived from that one start time.

2835 C.1.1 Academic Scheduling example



2836

2837

Figure 5: Classroom Scheduling Example

2838 A college campus uses two schedules to schedule its buildings. In Schedule 1, classes start on the hour,
2839 and follow one after another; each class starts on the hour. In the second schedule, each class lasts an
2840 hour and a quarter, and there is a fifteen minute gap between classes; classes start on the half hour. On
2841 many campuses, the sequence in Schedule 1 may describe classes taught on Monday, Wednesday, and
2842 Friday. Schedule 2 may describe classes taught on Tuesday and Thursday.

2843 The registrar's office knows some key facts about each classroom, including whether it hosts a class
2844 during a particular period, and the number of students that will be in that class. The college wishes to
2845 optimize the provision of building services for each class. Such services may include adequate ventilation
2846 and comfortable temperatures to assure alert students. Other services may ensure that the classroom
2847 projection systems and A/V support services are warmed up in advance of a class, or powered off when a
2848 classroom is vacant.

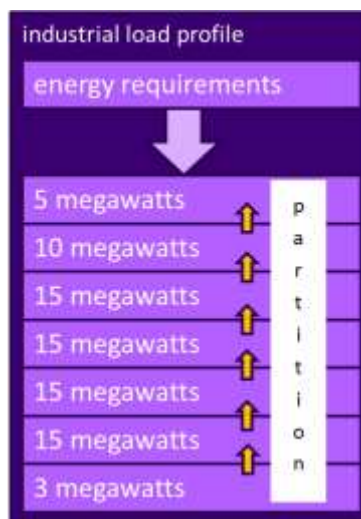
2849 Although most classes meet over typical schedule for the week (M-W-F or Tu-Th), some classes may not
2850 meet on Friday, or may have a tutorial section one day a week. The registrar's system, ever mindful of
2851 student privacy, shares only minimal information with the building systems such as how many students
2852 will be supported.

2853 The Registrar's system schedule building systems using the Calendar Gluon (registrar's information) and
2854 the student counts for each interval, and schedules the Sequence in classroom schedule 1 three days a
2855 week for the next 10 weeks. The Registrar's system also schedules the sequence in classroom schedule
2856 2 two days a week, also for 10 weeks.

2857 This example demonstrates a system (A) that offers services using either of two sequences. Another
2858 business system (B) with minimal knowledge of how (A) works determines the performance requirements
2859 for (A). The business system (B) communicates these expectations are by scheduling the Sequences
2860 offered by (A).

2861 C.1.2 Market Performance schedule

2862 A factory relies on an energy-intensive process which is performs twice a year for eight weeks. The
2863 factory has some flexibility about scheduling the process; it can perform the work in either the early
2864 morning or the early evening; it avoids the afternoon when energy costs are highest. The factory works up
2865 a detailed profile of when it will need energy to support this process.



2866
2867 *Figure 6: Daily Load Profile for Market Operations Example*

2868 Factory management has decided that they want to use only renewable energy products for this process.
2869 They approach two regional wind farms with the intent of making committed purchases of wind energy.
2870 The wind farms consider their proposals taking into account the seasonal weather forecasts they use to
2871 project their weather capacity, and considering the costs that may be required to buy additional wind
2872 energy on the spot market to make up any shortfalls.

2873 Each energy supplier submits of the same sequence, a schedule, i.e. a daily starting time, and a price for
2874 the season's production. After considering the bids, and other internal costs of each proposal, the factory
2875 opts to accept a contract for the purchase of a fixed load profile (Partition), using the evening wind
2876 generation from one of the suppliers. This contract specifies Schedules of load purchases (starting data
2877 and time for the sequence) for each day.

2878

Revision History

Revision	Date	Editor	Changes Made
1.0 WD 01	2010-03-11	Toby Considine	Initial document, largely derived from Charter
1.0 WD 02	2010-03-30	Toby Considine	Straw-man assertion of elements, components to push conversation
1.0 WD 03	2010-04-27	Toby Considine	Cleaned up Elements, added [XPOINTER] use, xs:duration elements
1.0 WD 04	2010-05-09	Toby Considine	Aligned Chapter 4 with the vAlarm and vToDo objects.
1.0 WD 05	2010-05-18	Toby Considine	Responded to comments, added references, made references to [XCAL] more consistent,
1.0 WD 06	2010-05-10	Toby Considine	Responded to comments from CalConnect, mostly constancy of explanations
1.0 WD 07	2010-07-28	Toby Considine	Incorporated input from informal public review, esp. SGIP PAP04. Firmed up relationships between scheduled objects
1.0 WD 08	2010-08-07	Toby Considine	Aligned with Interval / Partition / Sequence language. Reduced performance characteristics to before / after durations.
1.0 WD 09	2010-08-15	Toby Considine	Formalized Attachment section and rolled Performance into the Attachment. Created RelatedComponent object. Added CalWS Outline to specification. Removed SOOP section
1.0 WD 10	2010-08-28	Toby Considine, Benoit Lepeuple	Updated Time Stamp section Added background Appendices Incorporated Association language to replace RelatedComponent Recast examples to show inheritance, remove inconsistencies
1.0 WD 11	2010-09-11	Toby Considine	Traceability Release in support of a re-shuffling of the document. Sections 3, 4 were re-shuffled to create: 3: Interval / Relationships / Time Stamps 4: Performance / Attachments 5: Associations & Inheritance Also, changed all associations to Gluons. No paragraphs have been changed, just shuffled, changes accepted, to create clean base for editing
1.0 WD 12	2010-09-14	Toby Considine Dave Thewlis	Edits for clarity and flow following changes in WD11, updated examples based upon XSD artifacts. Adding final contribution from CalConnect for Services.

2881
2882
