



WS-Calendar Version 1.0

Working Draft 12

13 September 2010

Specification URIs:

This Version:

- <http://docs.oasis-open.org/WS-Calendar/v1.0/wd12/WS-Calendar-1.0-spec-wd-12.pdf>
- <http://docs.oasis-open.org/WS-Calendar/v1.0/wd12/WS-Calendar-1.0-spec-wd-12.html>
- <http://docs.oasis-open.org/WS-Calendar/v1.0/wd12/WS-Calendar-1.0-spec-wd-12.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.....	9
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.....	43
147	7	Calendar Services.....	44
148	7.1	Overview of the protocol.....	44
149	7.1.1	Calendar Object Resources.....	44
150	7.1.2	Timezone information.....	44
151	7.1.3	Issues not addressed by this specification.....	45
152	7.1.4	CalWS Glossary.....	45

153	7.2 Error conditions.....	46
154	7.2.1 Example: error with CalDAV error condition	46
155	8 Properties and link relations	47
156	8.1 Property and relation-type URIs	47
157	8.2 supported-features property.	47
158	8.3 max-attendees-per-instance	47
159	8.4 max-date-time	47
160	8.5 max-instances.....	47
161	8.6 max-resource-size	47
162	8.7 min-date-time	48
163	8.8 description.....	48
164	8.9 timezone-service relation.....	48
165	8.10 principal-home relation.	48
166	8.11 current-principal-freebusy relation.	48
167	8.12 principal-freebusy relation.....	48
168	8.13 child-collection relation.	48
169	8.14 created link property	49
170	8.15 last-modified property	49
171	8.16 displayname property	49
172	8.17 timezone property	49
173	8.18 owner property	49
174	8.19 collection link property	49
175	8.20 calendar-collection link property	49
176	8.21 CalWS:privilege-set XML element.....	50
177	9 Retrieving Collection and Service Properties.....	51
178	9.1 Request parameters	51
179	9.2 Responses:	51
180	9.3 Example - retrieving server properties:.....	51
181	10 Creating Calendar Object Resources.....	53
182	10.1 Request parameters	53
183	10.2 Responses:	53
184	10.3 Preconditions for Calendar Object Creation	53
185	10.4 Example - successful POST:.....	54
186	10.5 Example - unsuccessful POST:.....	54
187	11 Retrieving resources.....	55
188	11.1 Request parameters	55
189	11.2 Responses:	55
190	11.3 Example - successful fetch:	55
191	11.4 Example - unsuccessful fetch:	55
192	12 Updating resources	56
193	12.1 Responses:	56
194	13 Deletion of resources.....	58
195	13.1 Delete for Collections.....	58
196	13.2 Responses:	58
197	14 Querying calendar resources	59

198	14.1 Limiting data returned	59
199	14.2 Pre/postconditions for calendar queries	59
200	14.3 Example: time range limited retrieval	59
201	15 Free-busy queries.....	64
202	15.1 ACCEPT header	64
203	15.2 URL Query Parameters	64
204	15.2.1 start.....	64
205	15.2.2 end.....	65
206	15.2.3 period.....	65
207	15.2.4 account.....	65
208	15.3 URL parameters - notes	65
209	15.4 HTTP Operations	65
210	15.5 Response Codes	65
211	15.6 Examples	66
212	16 Conformance	69
213	A. Acknowledgements	70
214	B. An Introduction to Internet Calendaring.....	71
215	B.1 icalendar	71
216	B.1.1 History	71
217	B.1.2 Data model.....	71
218	B.1.3 Scheduling	72
219	B.1.4 Extensibility	72
220	B.2 Calendar data access and exchange protocols	72
221	B.2.1 Internet Calendar Subscriptions.....	72
222	B.2.2 CalDAV	72
223	B.2.3 ActiveSync/SyncML	73
224	B.2.4 CalWS.....	73
225	B.2.5 iSchedule	73
226	B.3 References	73
227	C. Overview of WS-Calendar, its Antecedents and its Use	74
228	C.1 Scheduling Sequences	75
229	C.1.1 Academic Scheduling example.....	75
230	C.1.2 Market Performance schedule.....	76
231	Revision History	77
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 56

258 Example 17: Unsuccessful Update 56

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 [Calendar Resource Schema C. Joy, C. Daboo, M Douglas, *Schema for representing
306 resources for calendaring and scheduling services*, [http://tools.ietf.org/html/draft-cal-resource-
307 schema-00](http://tools.ietf.org/html/draft-cal-resource-schema-00), (Internet-Draft), April 2010.

308 **FreeBusy Read URL** E York. *Freebusy read URL*,
309 <http://www.calconnect.org/pubdocs/CD0903%20Freebusy%20Read%20>
310 [URL%20V1.0.pdf](http://www.calconnect.org/pubdocs/CD0903%20Freebusy%20Read%20)

311 RFC2119].

312 1.2 Normative References

313 **Calendar Resource Schema** C. Joy, C. Daboo, M Douglas, *Schema for representing*
314 *resources for calendaring and scheduling services*,
315 <http://tools.ietf.org/html/draft-cal-resource-schema-00>, (Internet-Draft),
316 April 2010.

317 **FreeBusy Read URL** E York. *Freebusy read URL*,
318 <http://www.calconnect.org/pubdocs/CD0903%20Freebusy%20Read%20>
319 [URL%20V1.0.pdf](http://www.calconnect.org/pubdocs/CD0903%20Freebusy%20Read%20)

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

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

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

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

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

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

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

337 **RFC5546** C. Daboo *iCalendar Transport-Independent Interoperability Protocol*
338 *(iTIP)*, <http://www.ietf.org/RFC/RFC5546.txt>, IETF RFC5546, January
339 1999.

340 **SOA-RM** OASIS Standard, *Reference Model for Service Oriented Architecture 1.0*,
341 October 2006.

342 <http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.pdf>

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

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

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

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

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

- 355 **XML SCHEMA** PV Biron, A Malhotra, *XML Schema Part 2: Datatypes Second Edition*,
356 <http://www.w3.org/TR/xmlschema-2/> October 2004.
- 357 **XRD** OASIS XRI Committee Draft 01, *Extensible Resource Descriptor (XRD)*
358 *Version 1.0*, <http://docs.oasis-open.org/xri/xrd/v1.0/cd01/xrd-1.0-cd01.pdf>
359 October 2009.

360 1.3 Non-Normative References

- 361 **NIST Framework and Roadmap for Smart Grid Interoperability Standards**, Office of the
362 National Coordinator for Smart Grid Interoperability, Release 1.0, NIST
363 Special Publication 1108,
364 http://www.nist.gov/public_affairs/releases/upload/smartgrid_interoperability_final.pdf January 2010.
- 365 **NAESB Smart Grid Requirements** (*dunno what reference I need here*)
- 366
- 367
- 368 **REST** T Fielding, *Architectural Styles and the Design of Network-based*
369 *Software Architectures*,
370 <http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>.
- 371 **Time Zone Recommendations**, CalConnect, *CalConnect EDST (Extended Daylight*
372 *Savings Time) Reflections and Recommendations*, Version: 1.1,
373 <http://www.calconnect.org/pubdocs/CD0707%20CalConnect%20EDST%20Reflections%20and%20Recommendations%20V1.1.pdf>
374 2007-10-04
- 375
- 376 **Time Zone Service**, M Douglas, C Daboo, *Timezone Service Protocol*, Draft RFC,IETF,
377 <http://datatracker.ietf.org/doc/draft-douglass-timezone-service/>
378 2007-07-05
- 379

380 1.4 Naming Conventions

- 381 This specification follows some naming conventions for artifacts defined by the specification, as follows:
- 382 For the names of elements and the names of attributes within XSD files, the names follow the
383 CamelCase¹ convention, with all names starting with a lower case letter, eg
- 384 `<element name="componentType" type="WS-Calendar:ComponentType"/>`
- 385 For the names of types within XSD files, the names follow the CamelCase convention with all names
386 starting with an upper case letter, e.g.,
- 387 `<complexType name="ComponentService">`
- 388 For the names of intents, the names follow the CamelCase convention, with all names starting with a
389 lower case letter, EXCEPT for cases where the intent is to represent an established acronym, in which
390 case the entire name follows the usage of the established acronym.
- 391 An example of an intent which references an acronym is the "SOAP" intent.

392 1.5 Architectural References

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

¹ Common term - see Wikipedia for explanation. Actually lower Camel Case. Also used in CIM reference guides.

396 2 Overview of WS-Calendar

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

403 2.1 Approach taken by the WS-Calendar Technical Committee

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

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

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

418 2.2 Scheduling Service Performance

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

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

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

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

432 A request for remote service performance is a request for specific real world effects. Consider two service
433 providers that offer the same service. One must start planning an hour or more in advance. The second
434 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 1.2

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

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

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

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

445 **2.2.1 Which Time? UCT vs. Local Time**

446 When 2 or more parties attempt to agree on a time - e.g., for a meeting, or when to provide a service,
447 they agree to start at a particular instant of time UTC. They agree on that instant in time by converting
448 from local time, e.g., they want a meeting to start at 13:00 Eastern, 18:00 UK. Our lives and the use of
449 services are bound by local time not by UTC. To humans local time is the invariant and UTC is mapped
450 on to it. If a government messes with the rules we adjust the mappings and we shift the UTC time. We still
451 want to meet at 13:00 local or have the heating start at 0700.

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

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

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

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

467 **2.3 Overview of This Document**

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

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

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

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

482 3 Intervals, Temporal Relations, and Sequences

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

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

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

494 3.1 Core Semantics derived from [XCAL]

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

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

500 3.1.1 Time

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

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

515 Normative information on [ISO 8601] is found in section 1.2.

516 The iCalendar Components (VComponents)

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

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

524 3.2 Intervals

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

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

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

535 *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.

536 1.1.1 Intervals: the Basic Time Segment

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

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

545 *Table 3-2: VTODO elements 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.

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

549

Example 1: An Interval

550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567

```

<components>
<vtodo>
  <properties>
    <x-wscalendartype>Interval</x-wscalendartype>
    <uid>
      <text>00959BC664CA650E933C892C@example.com</text>
    </uid>
    <description>
      <text>Sample Contract</text>
    </description>
    <duration>
      <duration>
        <duration>T10H</duration>
      </duration>
    </duration>
  </properties>
</vtodo>
</components>

```

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

570 3.3 Temporal Relations between Intervals

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

575
576
577

```

<relationship>
  <uid>aaaaaaaa1</uid>
  <reltype>PARENT</reltype>

```

578

```
</relationship>
```

579
580
581

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

582

Table 3-3: Temporal Relationships in WS-Calendar

Temporal Relationship	Short Form	Definition
FINISHSTART	FS	As soon as the related Component finishes, this interval begins.
FINISHFINISH	FF	Used without gap when to 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.
STARTFINISH	SF	This component must Finish before the related component starts.
STARTSTART	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.

583
584
585

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

586

Table 3-4: Elements of a 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

587
588

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

589

Example 2: Temporal Relationship

590
591
592
593
594
595
596
597
598
599
600

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

601
602
603

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

604

Example 3: Temporal Relationship with and without Gap

605
606
607
608
609
610
611
612
613
614
615

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

616 Leaving out the optional Gap element, we have:

617
618
619
620
621
622

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

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

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

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

630 3.4 Sequences: Combining Intervals

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

634 *Table 3-5: Introducing the Sequence*

635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658

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

```

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

```

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

702 3.4.1 Scheduling a Sequence

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

705 *Example 4: A Scheduled Sequence*

```

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

```

```

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

```

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

744 3.5 Alarms

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

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

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

758 3.6 Time Stamps

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

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

766 *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.

767 **3.6.1 Time Stamp Realm Discussion**

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

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

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

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

784 **4 Service Characteristics: Attachments &**
 785 **Performance**

786 **4.1 Services and Service Characteristics**

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

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

793 **4.1.1 Attachments**

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

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

800 *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.

801 When a WS-Calendar reference uses an external reference to specify a service, that reference is an
 802 object of the type [XPOINTER] (see section 1.2)..[XPOINTER] is a general purpose URI and XML
 803 traversal standard. This [XPOINTER] object is in the named data element “Reference.”

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

```

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

```

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

```

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

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

```

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

```

840

841 4.1.2 Specifying Timely Performance

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

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

847 *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.

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

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

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

854 *Example 7: Performance Component*

```

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

```

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

885 Generally, the implementer should refrain from expressing unnecessary or redundant performance
886 characteristics.

887 4.1.3 Combining Service and Performance

888 Services, references and performance each appear in the ATTACH element of the iCalendar
889 components.

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

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

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

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

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

952

5 Inheritance and Entry Points: Calendar Gluons

953

5.1.1 Calendar Gluons

954

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.

958

959

960

961

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.

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

969 5.1.2 Calendar Gluons and Sequences

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

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

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

```
975 <components>  
976 <x- calendargluon>  
977   <properties>  
978     <x-wscalendartype>CalendarGluon</x-wscalendartype>  
979     <xcal:uid>  
980       <xcal:text>00959BC664CA650E933C892C@example.com</xcal:text>  
981     </xcal:uid>  
982     <xcal:description>  
983       <xcal:text> Calendar Gluon with sequence </xcal:text>  
984     </xcal:description>  
985     <xcal:comment>  
986       <xcal:text> creates common performance expectations (+/- 1 second)  
987         for the entire sequence. Also sets common duration (15  
988         minutes) for all members of the sequence, No interval may end  
989         after its scheduled end-time </xcal:text>  
990     </xcal:comment>  
991     <xcal:duration>  
992       <xcal:duration>T15M</xcal:duration>  
993     </xcal:duration>  
994     <attach>  
995       <performance>  
996         <properties>  
997           <endbeforetolerance>  
998             <duration>  
999               <duration>T1S</duration>  
1000             </duration>  
1001           </endbeforetolerance>  
1002           <endaftertolerance>  
1003             <duration>  
1004               <duration>T1S</duration>  
1005             </duration>  
1006           </endaftertolerance>  
1007         </properties>  
1008       </performance>  
1009     </attach>  
1010     <xcal:related-to>  
1011       <xcal:parameters>  
1012         <xcal:reltype>PARENT</xcal:reltype>  
1013       </xcal:parameters>  
1014       <xcal:text>10959BC664CA650E933C892C@example.com </xcal:text>  
1015     </xcal:related-to>  
1016   </properties>  
1017 </x- calendargluon>  
1018 <vtodo>  
1019   <properties>  
1020     <x-wscalendartype>Interval</x-wscalendartype>
```

```

1021     <xcal:uid>
1022         <xcal:text>10959BC664CA650E933C892C@example.com</xcal:text>
1023     </xcal:uid>
1024     <xcal:description>
1025         <xcal:text>First Interval in Sequence</xcal:text>
1026     </xcal:description>
1027     <xcal:summary>
1028         <xcal:text>Inherits all performance from Gluon as well
1029             As the duration</xcal:text>
1030     </xcal:summary>
1031     <attach>
1032         <artifact>
1033             <emix-wip>
1034                 <price>8.45</price>
1035                 <quantity>4200</quantity>
1036             </emix-wip>
1037         </artifact>
1038     </attach>
1039 </properties>
1040 </vtodo>
1041 <vtodo>
1042     <properties>
1043         <x-wscalendartype>Interval</x-wscalendartype>
1044         <xcal:uid>
1045             <xcal:text>20959BC664CA650E933C892C@example.com</xcal:text>
1046         </xcal:uid>
1047         <xcal:description>
1048             <xcal:text>Second Interval in Sequence</xcal:text>
1049         </xcal:description>
1050         <xcal:summary>
1051             <xcal:text>Inherits all performance from Gluon, follows
1052                 Finish of Interval 1, inherits duration</xcal:text>
1053         </xcal:summary>
1054         <attach>
1055             <artifact>
1056                 <emix-wip>
1057                     <price>8.49</price>
1058                     <quantity>4500</quantity>
1059                 </emix-wip>
1060             </artifact>
1061         </attach>
1062         <x-wscalendarrelation>
1063             <temporalrelationshiptype>finishtostart</temporalrelationshiptype>
1064             <relatedto>
1065                 <uid>10959BC664CA650E933C892C@example.com</uid>
1066             </relatedto>
1067             <gap>
1068                 <xcal:duration>
1069                     <xcal:duration>T0M</xcal:duration>
1070                 </xcal:duration>
1071             </gap>
1072         </x-wscalendarrelation>
1073     </properties>
1074 </vtodo>
1075 <vtodo>
1076     <properties>
1077         <x-wscalendartype>Interval</x-wscalendartype>
1078         <xcal:uid>
1079             <xcal:text>30959BC664CA650E933C892C@example.com</xcal:text>
1080         </xcal:uid>
1081         <xcal:description>
1082             <xcal:text>Third Interval in Sequence</xcal:text>
1083         </xcal:description>
1084         <xcal:summary>

```

```

1085         <xcal:text>Inherits all performance from Gluon, follows
1086             Finish of Interval 2, overrides duration</xcal:text>
1087     </xcal:summary>
1088     <attach>
1089         <artifact>
1090             <emix-wip>
1091                 <price>7.45</price>
1092                 <quantity>6000</quantity>
1093             </emix-wip>
1094         </artifact>
1095     </attach>
1096     <xcal:duration>
1097         <xcal:duration>T30M</xcal:duration>
1098     </xcal:duration>
1099     <x-wscalendarrelation>
1100         <temporalrelationshipstype>finishtostart</temporalrelationshipstype>
1101         <relatedto>
1102             <uid>20959BC664CA650E933C892C@example.com</uid>
1103         </relatedto>
1104         <gap>
1105             <xcal:duration>
1106                 <xcal:duration>T5M</xcal:duration>
1107             </xcal:duration>
1108         </gap>
1109     </x-wscalendarrelation>
1110 </properties>
1111 </vtodo>
1112 </components>

```

1113 Note that the performance expectations, identical for each interval, have moved into the Calendar Gluon.
1114 Not also that while the duration for all intervals in the partition is set in the Calendar Gluon, interval 3
1115 overrides that with a half hour duration assigned locally. The Calendar Gluon happens to related to the
1116 first Interval in the sequence; there are specific use cases (discussed below) which require it to be linked
1117 to other Intervals.

1118 5.1.3 Inheritance rules for Calendar Gluons

1119 In general, the rules that anything specified in the Parent Calendar Gluon applies to each Child. The
1120 Parent of an Interval in a Sequence is parent to all Intervals in the Sequence. As a Sequence creates
1121 single temporal relationship, assigning a start time (dtstart) to any Interval allows the starting time to be
1122 computed for any of them.

1123 *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.

1124 5.1.4 Optimizing the expression of a Partition

1125 Partitions are Sequences with consecutive Intervals. Communication of a Partition can be further
 1126 optimized by bringing the relationship into the Calendar Gluon. Notice that while the type of the
 1127 relationship is defined in the Calendar Gluon, the Temporal Relation for each Interval must still be
 1128 expressed within the Interval.

1129 *Example 11: Partition with Duration and Performance defined in the Calendar Gluon*

```

1130 <components>
1131 <x-calendargluon>
1132   <properties>
1133     <x-wscalendartype>CalendarGluon</x-wscalendartype>
1134     <xcal:uid>
1135       <xcal:text>00959BC664CA650E933C892C@example.com</xcal:text>
1136     </xcal:uid>
1137     <xcal:description>
1138       <xcal:text>Calendar Gluon for energy markets with consecutive
1139         Identical intervals</xcal:text>
1140     </xcal:description>
1141     <xcal:comment>
1142       <xcal:text> creates common performance expectations that the
1143         granularity for measuring all Intervals is (+/- 1 second)
1144         Also sets common duration (15 minutes)for all members of
1145         the sequence). Each interval in the partitions begins
1146         immediately after its predecessor finishes. </xcal:text>
1147     </xcal:comment>
1148     <xcal:duration>
1149       <xcal:duration>T15M</xcal:duration>
1150     </xcal:duration>
1151     <attach>
1152       <artifact>
1153         <emix-wip>PRODUCT SPECIFICATION UNDEFINED</emix-wip>
1154       </artifact/>
1155       <performance>
1156         <properties>
1157           <granularity>
1158             <duration>
1159               <duration>T1S</duration>
1160             </duration>
1161           </granularity>
1162         </properties>
1163       </performance>
1164     </attach>
1165     <x-wscalendarrelation>
1166       <temporalrelationshipiptye>finishtostart</temporalrelationshipiptye>
1167     <gap>

```

```

1168         <xcal:duration>
1169             <xcal:duration>T0S</xcal:duration>
1170         </xcal:duration>
1171     </gap>
1172 </x-wscalendarrelation>
1173 <xcal:related-to>
1174     <xcal:parameters>
1175         <xcal:reltype>PARENT</xcal:reltype>
1176     </xcal:parameters>
1177     <xcal:text>10959BC664CA650E933C892C@example.com </xcal:text>
1178 </xcal:related-to>
1179 </properties>
1180 </x-calendargluon>
1181 <vtodo>
1182     <properties>
1183         <x-wscalendartype>Interval</x-wscalendartype>
1184         <xcal:uid>
1185             <xcal:text>10959BC664CA650E933C892C@example.com</xcal:text>
1186         </xcal:uid>
1187         <xcal:description>
1188             <xcal:text>First Interval in Sequence</xcal:text>
1189         </xcal:description>
1190         <xcal:summary>
1191             <xcal:text>Inherits all performance from Gluon as well
1192                 As the duration and the Temporal Relation</xcal:text>
1193         </xcal:summary>
1194         <attach>
1195             <artifact>
1196                 <emix-wip>
1197                     <price>8.45</price>
1198                     <quantity>4200</quantity>
1199                 </emix-wip>
1200             </artifact/>
1201         </attach>
1202     </properties>
1203 </vtodo>
1204 <vtodo>
1205     <properties>
1206         <x-wscalendartype>Interval</x-wscalendartype>
1207         <xcal:uid>
1208             <xcal:text>20959BC664CA650E933C892C@example.com</xcal:text>
1209         </xcal:uid>
1210         <xcal:description>
1211             <xcal:text>Second Interval in Sequence</xcal:text>
1212         </xcal:description>
1213         <attach>
1214             <artifact>
1215                 <emix-wip>
1216                     <price>8.49</price>
1217                     <quantity>4500</quantity>
1218                 </emix-wip>
1219             </artifact/>
1220         </attach>
1221         <x-wscalendarrelation>
1222             <relatedto>
1223                 <uid>10959BC664CA650E933C892C@example.com</uid>
1224             </relatedto>
1225         </x-wscalendarrelation>
1226     </properties>
1227 </vtodo>
1228 <vtodo>
1229     <properties>
1230         <x-wscalendartype>Interval</x-wscalendartype>
1231         <xcal:uid>

```



```

1232     <xcal:text>30959BC664CA650E933C892C@example.com</xcal:text>
1233 </xcal:uid>
1234 <xcal:description>
1235     <xcal:text>Third Interval in Sequence</xcal:text>
1236 </xcal:description>
1237 <xcal:summary>
1238     <xcal:text>Inherits all performance from Gluon, follows
1239     Finish of Interval 2, overrides duration</xcal:text>
1240 </xcal:summary>
1241 <attach>
1242     <artifact>
1243         <emix-wip>
1244             <price>7.45</price>
1245             <quantity>6000</quantity>
1246         </emix-wip>
1247     </artifact>
1248 </attach>
1249 <x-wscalendarrelation>
1250     <relatedto>
1251         <uid>20959BC664CA650E933C892C@example.com</uid>
1252     </relatedto>
1253 </x-wscalendarrelation>
1254 </properties>
1255 </vtodo>
1256 </components>

```

1257 This Partition shows a school schedule in which classes start one hour apart. Each service is performed
1258 for 50 minutes, and there is a 10 minute gap between each as students move between classes. Classes
1259 may not begin before the schedule, but they may start up to five minutes late.

1260 Stripped of all annotations, this can be expressed as follows:

1261 *Example 12: Partition without annotations*

```

1262 <components>
1263 <x-calendargluon>
1264     <properties>
1265         <x-wscalendarstype>CalendarGluon</x-wscalendarstype>
1266         <xcal:uid>
1267             <xcal:text>00959BC664CA650E933C892C@example.com</xcal:text>
1268         </xcal:uid>
1269         <xcal:duration>
1270             <xcal:duration>T50M</xcal:duration>
1271         </xcal:duration>
1272         <attach>
1273             <artifact>
1274                 <classroom>demonstration specification</classroom>
1275             </artifact>
1276         </attach>
1277         <x-wscalendarrelation>
1278             <temporalrelationshipstype>finishtostart</temporalrelationshipstype>
1279             <gap>
1280                 <xcal:duration>
1281                     <xcal:duration>T10M</xcal:duration>
1282                 </xcal:duration>
1283             </gap>
1284         </x-wscalendarrelation>
1285         <xcal:related-to>
1286             <xcal:parameters>
1287                 <xcal:reltype>PARENT</xcal:reltype>
1288             </xcal:parameters>
1289             <xcal:text>10959BC664CA650E933C892C@example.com </xcal:text>
1290         </xcal:related-to>
1291     </properties>
1292 </x-calendargluon>

```

```

1293 <vtodo>
1294   <properties>
1295     <x-wscalendartype>Interval</x-wscalendartype>
1296     <xcal:uid>
1297       <xcal:text>10959BC664CA650E933C892C@example.com</xcal:text>
1298     </xcal:uid>
1299     <attach>
1300       <artifact>
1301         <classroom><students>48</students></classroom>
1302       <artifact/>
1303     </attach>
1304   </properties>
1305 </vtodo>
1306 <vtodo>
1307   <properties>
1308     <x-wscalendartype>Interval</x-wscalendartype>
1309     <xcal:uid>
1310       <xcal:text>20959BC664CA650E933C892C@example.com</xcal:text>
1311     </xcal:uid>
1312     <attach>
1313       <artifact>
1314         <classroom><students>65</students></classroom>
1315       <artifact/>
1316     </attach>
1317     <x-wscalendarrelation>
1318       <relatedto>
1319         <uid>10959BC664CA650E933C892C@example.com</uid>
1320       </relatedto>
1321     </x-wscalendarrelation>
1322   </properties>
1323 </vtodo>
1324 <vtodo>
1325   <properties>
1326     <x-wscalendartype>Interval</x-wscalendartype>
1327     <xcal:uid>
1328       <xcal:text>30959BC664CA650E933C892C@example.com</xcal:text>
1329     </xcal:uid>
1330     <attach>
1331       <artifact>
1332         <classroom><students>34</students></classroom>
1333       <artifact/>
1334     </attach>
1335     <x-wscalendarrelation>
1336       <relatedto>
1337         <uid>20959BC664CA650E933C892C@example.com</uid>
1338       </relatedto>
1339     </x-wscalendarrelation>
1340   </properties>
1341 </vtodo>
1342 <components>

```

1343 A sequence can also be scheduled in the Calendar Gluon.

1344 *Example 13: A Scheduled Sequence showing Temporal Relationship Inheritance*

```

1345 <components>
1346 <x-calendargluon>
1347   <properties>
1348     <x-wscalendartype>CalendarGluon</x-wscalendartype>
1349     <xcal:uid>
1350       <xcal:text>00959BC664CA650E933C892C@example.com</xcal:text>
1351     </xcal:uid>
1352     <dtstart>2010-09-11 T00:15</dtstart>
1353   <attach>
1354     <artifact>

```

```

1355         <classroom>demonstration specification</classroom>
1356         <artifact/>
1357     </attach>
1358     <xcal:related-to>
1359         <xcal:parameters>
1360             <xcal:reltype>PARENT</xcal:reltype>
1361         </xcal:parameters>
1362         <xcal:text>10959BC664CA650E933C892C@example.com </xcal:text>
1363     </xcal:related-to>
1364 </properties>
1365 </x-calendargluon>
1366 <vtodo>
1367     <properties>
1368         <x-wscalendartype>Interval</x-wscalendartype>
1369         <xcal:uid>
1370             <xcal:text>10959BC664CA650E933C892C@example.com</xcal:text>
1371         </xcal:uid>
1372         <xcal:description>
1373             <xcal:text>First Interval in Sequence</xcal:text>
1374         </xcal:description>
1375         <dtstart>2010-09-11T13:00</dtstart>
1376         <xcal:duration>
1377             <xcal:duration>T1H</xcal:duration>
1378         </xcal:duration>
1379     </properties>
1380 </vtodo>
1381 <vtodo>
1382     <properties>
1383         <x-wscalendartype>Interval</x-wscalendartype>
1384         <xcal:uid>
1385             <xcal:text>20959BC664CA650E933C892C@example.com</xcal:text>
1386         </xcal:uid>
1387         <xcal:description>
1388             <xcal:text>Second Interval in Sequence</xcal:text>
1389         </xcal:description>
1390         <xcal:duration>
1391             <xcal:duration>T15M</xcal:duration>
1392         </xcal:duration>
1393         <x-wscalendarrelation>
1394             <temporalrelationshipi>finishtostart</temporalrelationshipi>
1395             <relatedto>
1396                 <uid>10959BC664CA650E933C892C@example.com</uid>
1397             </relatedto>
1398         </x-wscalendarrelation>
1399     </properties>
1400 </vtodo>
1401 </components>

```

1402 5.1.5 Mixed Inheritance of Start Time

1403 A Sequence has not been scheduled until it has both a start time and a start date. Start time and date
1404 SHALL be expressed together when all components are in a single communication. Time and Date MAY
1405 be separated when the full sequence and schedule are created by reference.

1406 To illustrate this, here is the classroom scheduling Partition from Example 12, updated to include each
1407 day's school opening.

1408 *Example 14: Partition with Duration and Performance defined in the Calendar Gluon*

```

1409 <components>
1410 <x-calendargluon>
1411     <properties>
1412         <x-wscalendartype>CalendarGluon</x-wscalendartype>
1413         <xcal:uid>

```

```

1414         <xcal:text>
1415             90959BC664CA650E933C892C@invokingexample.com
1416         </xcal:text>
1417     </xcal:uid>
1418     <dtstart>2010-09-13T09:00</dtstart>
1419     <xcal:related-to>
1420         <xcal:parameters>
1421             <xcal:reltype>PARENT</xcal:reltype>
1422         </xcal:parameters>
1423         <xcal:uri>http://scheduled.ws-calendar-service.com/classSchedule
1424         </xcal:uri>
1425     </xcal:related-to>
1426 </properties>
1427 </x-calendargluon>
1428 </components>

```

1429 Here, an external Calendar Gluon (above) makes reference to a published classroom schedule service:

```

1430 <x-calendargluon>
1431     <properties>
1432         <x-wscalendartype>CalendarGluon</x-wscalendartype>
1433         <xcal:uid>
1434             <xcal:text>http://scheduled.ws-calendar-service.com/classSchedule
1435             </xcal:text>
1436         </xcal:uid>
1437         <xcal:description>
1438             <xcal:text>MWF Classroom Schedule
1439             Identical intervals</xcal:text>
1440         </xcal:description>
1441         <xcal:comment>
1442             <xcal:text>Publishes a common classroom schedule for Monday,
1443             Wednesday, Friday for a semester at a school. Note that each
1444             day starts at 9:00</xcal:text>
1445         </xcal:comment>
1446         <dtstart>T09:00</dtstart>
1447         <xcal:duration>
1448             <xcal:duration>T50M</xcal:duration>
1449         </xcal:duration>
1450         <attach>
1451             <xcal:uri></xcal:uri>
1452         </attach>
1453         <x-wscalendarrelation>
1454             <temporalrelationshipi>type>finishtostart</temporalrelationshipi>
1455             <gap>
1456                 <xcal:duration>
1457                     <xcal:duration>T10M</xcal:duration>
1458                 </xcal:duration>
1459             </gap>
1460         </x-wscalendarrelation>
1461         <xcal:related-to>
1462             <xcal:parameters>
1463                 <xcal:reltype>PARENT</xcal:reltype>
1464             </xcal:parameters>
1465             <xcal:text>10959BC664CA650E933C892C@example.com </xcal:text>
1466         </xcal:related-to>
1467     </properties>
1468 </x-calendargluon>
1469 <vtodo>
1470     <properties>
1471         <x-wscalendartype>Interval</x-wscalendartype>
1472         <xcal:uid>
1473             <xcal:text>10959BC664CA650E933C892C@example.com</xcal:text>
1474         </xcal:uid>
1475         <attach>

```

```

1476         <artifact>
1477             <classroom><students>48</students></classroom>
1478         </artifact/>
1479     </attach>
1480 </properties>
1481 </vtodo>
1482 <vtodo>
1483     <properties>
1484         <x-wscalendartype>Interval</x-wscalendartype>
1485         <xcal:uid>
1486             <xcal:text>20959BC664CA650E933C892C@example.com</xcal:text>
1487         </xcal:uid>
1488         <attach>
1489             <artifact>
1490                 <classroom><students>65</students></classroom>
1491             </artifact/>
1492         </attach>
1493     <x-wscalendarrelation>
1494         <relatedto>
1495             <uid>10959BC664CA650E933C892C@example.com</uid>
1496         </relatedto>
1497     </x-wscalendarrelation>
1498 </properties>
1499 </vtodo>
1500 <vtodo>
1501     <properties>
1502         <x-wscalendartype>Interval</x-wscalendartype>
1503         <xcal:uid>
1504             <xcal:text>30959BC664CA650E933C892C@example.com</xcal:text>
1505         </xcal:uid>
1506         <attach>
1507             <artifact>
1508                 <classroom><students>34</students></classroom>
1509             </artifact/>
1510         </attach>
1511     <x-wscalendarrelation>
1512         <relatedto>
1513             <uid>20959BC664CA650E933C892C@example.com</uid>
1514         </relatedto>
1515     </x-wscalendarrelation>
1516 </properties>
1517 </vtodo>
1518 <components>
1519

```

1520 In the example above, a general purpose classroom calendar has been created and advertised with an
1521 URL. The class day always starts at 9:00. The referring Calendar Gluon scheduled a particular instance
1522 of this Sequence for Monday, September 13.

1523 This double inheritance, in which a Sequence inherits from a Calendar Gluon which inherits from a
1524 Calendar Gluon is a useful pattern for scheduling an advertised service.

1525 5.1.6 Other Scheduling Scenarios

1526 Sometimes, the invoker of a service is interested only in single Interval of the Sequence, but the entire
1527 Sequence is required. In the example below, the second Interval is advertised, i.e., the Calendar Gluon
1528 points to the second Interval. The first interval might be a required ramp-period, during which the
1529 underlying process is “warming up”, and which may bring some lesser service to market during that ramp
1530 time. The ramp-down time at the end is similarly fixed. The entire Service offering is represented by the
1531 exposed (it has a public URI) Calendar Gluon.

1532 *Example 15: Standard Sequence with Ramp-Up and Ramp Down*

```
1533 <components>
```

```

1534 <x- calendargluon>
1535   <properties>
1536     <x-wscalendartype>CalendarGluon</x-wscalendartype>
1537     <xcal:uid>
1538       <xcal:text>http://scheduled.ws-calendar-service.com/runcontract
1539       </xcal:text>
1540     </xcal:uid>
1541     <xcal:description>
1542       <xcal:text>Advertisement of schedule with ramp up and ramp down
1543       services.</xcal:text>
1544     </xcal:description>
1545     <xcal:comment>
1546       <xcal:text>Invokes second of three Intervals in the Sequence
1547       </xcal:text>
1548     </xcal:comment>
1549     <attach>
1550       <xcal:uri><emix-wip4></emix-wip></xcal:uri>
1551     </attach>
1552     <xcal:related-to>
1553       <xcal:parameters>
1554         <xcal:reltype>PARENT</xcal:reltype>
1555       </xcal:parameters>
1556       <xcal:text>20959BC664CA650E933C892C@example.com </xcal:text>
1557     </xcal:related-to>
1558   </properties>
1559   <xcal:duration>
1560     <xcal:duration>T6H</xcal:duration>
1561   </xcal:duration>
1562 </x- calendargluon>
1563 <vtodo>
1564   <properties>
1565     <x-wscalendartype>Interval</x-wscalendartype>
1566     <xcal:uid>
1567       <xcal:text>10959BC664CA650E933C892C@example.com</xcal:text>
1568     </xcal:uid>
1569     <xcal:description>
1570       <xcal:text>Ramp-Up Interval</xcal:text>
1571     </xcal:description>
1572     <xcal:summary>
1573       <xcal:text>Required as part of operations. Slowly increasing, yet
1574       fixed over-all, energy produced.
1575     </xcal:text>
1576     </xcal:summary>
1577     <xcal:duration>
1578       <xcal:duration>T45M</xcal:duration>
1579     </xcal:duration>
1580     <attach>
1581       <artifact>
1582         <emix-wip>describes ramp-up</emix-wip>
1583       </artifact>
1584     </attach>
1585   </properties>
1586 </vtodo>
1587 <vtodo>
1588   <properties>
1589     <x-wscalendartype>Interval</x-wscalendartype>
1590     <xcal:uid>
1591       <xcal:text>20959BC664CA650E933C892C@example.com</xcal:text>
1592     </xcal:uid>
1593     <xcal:description>

```

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

```

1594         <xcal:text>Run Interval</xcal:text>
1595     </xcal:description>
1596     <xcal:summary>
1597         <xcal:text>Inherits all performance from Gluon, follows
1598             Finish of Interval 2, overrides duration</xcal:text>
1599     </xcal:summary>
1600     <attach>
1601         <artifact>
1602             <emix-wip >Product Definition</emix-wip>
1603         </artifact>
1604     </attach>
1605     <x-wscalendarrelation>
1606         <relatedto>
1607             <uid>10959BC664CA650E933C892C@example.com</uid>
1608         </relatedto>
1609     </x-wscalendarrelation>
1610 </properties>
1611 </vtodo>
1612 <vtodo>
1613     <properties>
1614         <x-wscaleardtype>Interval</x-wscaleardtype>
1615         <xcal:uid>
1616             <xcal:text>30959BC664CA650E933C892C@example.com</xcal:text>
1617         </xcal:uid>
1618         <xcal:description>
1619             <xcal:text>Ramp-down Interval</xcal:text>
1620         </xcal:description>
1621         <xcal:summary>
1622             <xcal:text>Required as part of operations. Fixed time and fixed,
1623                 while diminishing, energy produced.
1624             </xcal:text>
1625         </xcal:summary>
1626         <xcal:duration>
1627             <xcal:duration>T30M</xcal:duration>
1628         </xcal:duration>
1629         <attach>
1630             <artifact>
1631                 <emix-wip>describes ramp-up</emix-wip>
1632             </artifact>
1633         </attach>
1634         <x-wscalendarrelation>
1635             <relatedto>
1636                 <uid>20959BC664CA650E933C892C@example.com</uid>
1637             </relatedto>
1638         </x-wscalendarrelation>
1639     </properties>
1640 </vtodo>

```

1641 When the service is scheduled, the time and duration are specified. The duration only applies to the
1642 Second Interval as all others have their duration explicitly specified.

```

1643 <components>
1644 <x-calendargluon>
1645     <properties>
1646         <x-wscaleardtype>CalendarGluon</x-wscaleardtype>
1647         <xcal:uid>
1648             <xcal:text>http://scheduled.ws-calendar-service.com/scheduleB
1649             </xcal:text>
1650         </xcal:uid>
1651         <xcal:description>
1652             <xcal:text>Advertisement of schedule with ramp up and ramp down
1653                 services.</xcal:text>
1654         </xcal:description>
1655         <xcal:comment>

```

```
1656         <xcal:text>Invokes second of three Intervals in the Sequence
1657         </xcal:text>
1658     </xcal:comment>
1659     <attach>
1660         <artifact>
1661             <emix-wip5>
1662                 <execute-price>15000</execute-price>
1663             </emix-wip>
1664         </artifact>
1665     </attach>
1666     <xcal:related-to>
1667         <xcal:parameters>
1668             <xcal:reltype>PARENT</xcal:reltype>
1669         </xcal:parameters>
1670         <xcal:text>http://scheduled.ws-calendar-service.com/runcontract
1671         </xcal:text>
1672     </xcal:related-to>
1673     <xcal:duration>
1674         <xcal:duration>T6H</xcal:duration>
1675     </xcal:duration>
1676 </properties>
1677 </x-calendargluon>
1678 </components>
```

1679 In this case, the specific interval is scheduled and a run time of 6 hours is specified for a price of \$15,000.

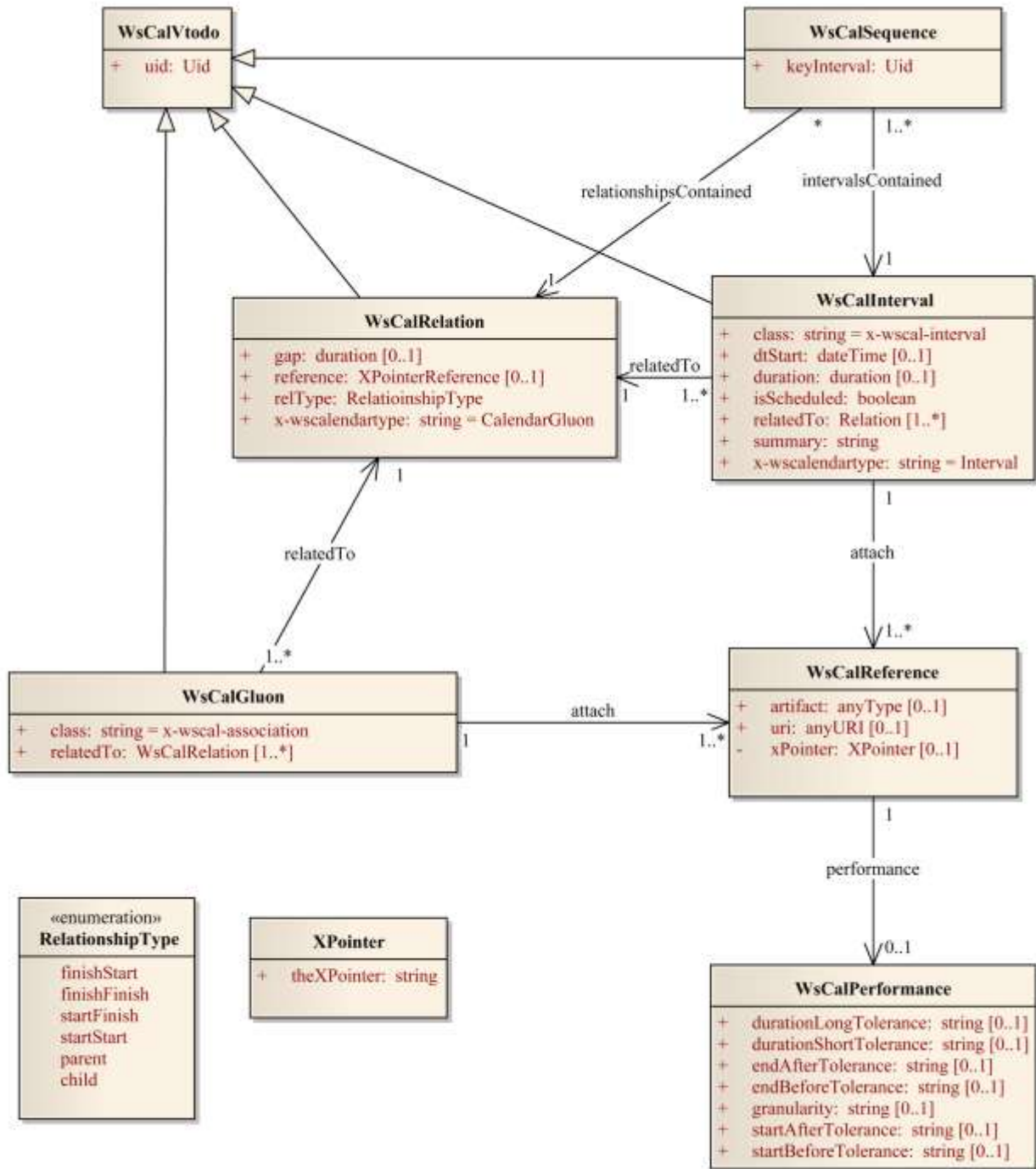
⁵ There is no EMIX-WIP specification. EMIX-WIP represents a generic energy market artifact, perhaps evocative of future specifications.

1680

6 WS-Calendar Models

1681

6.1 Abstract model for WS-Calendar Objects



1682

1683

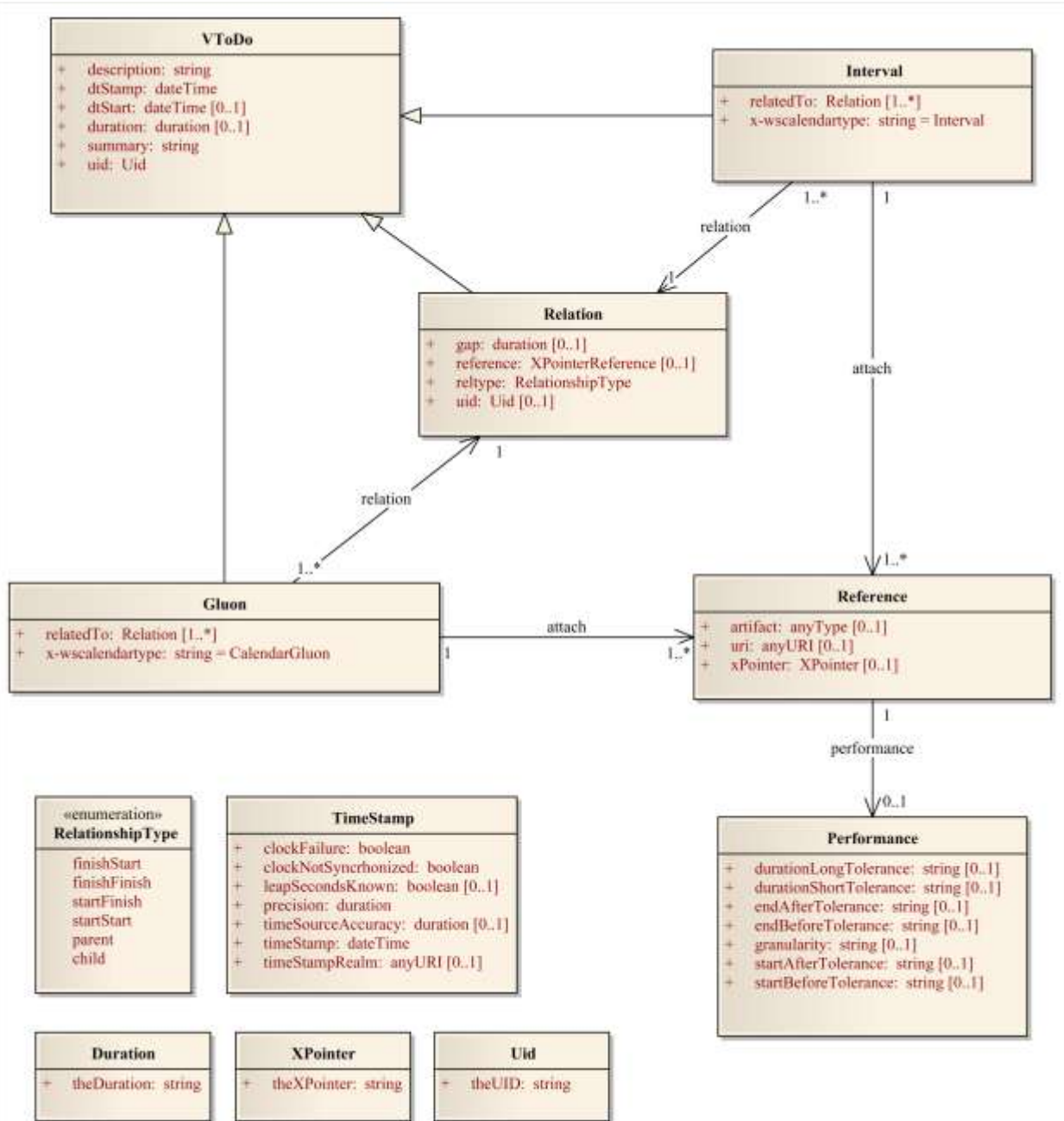
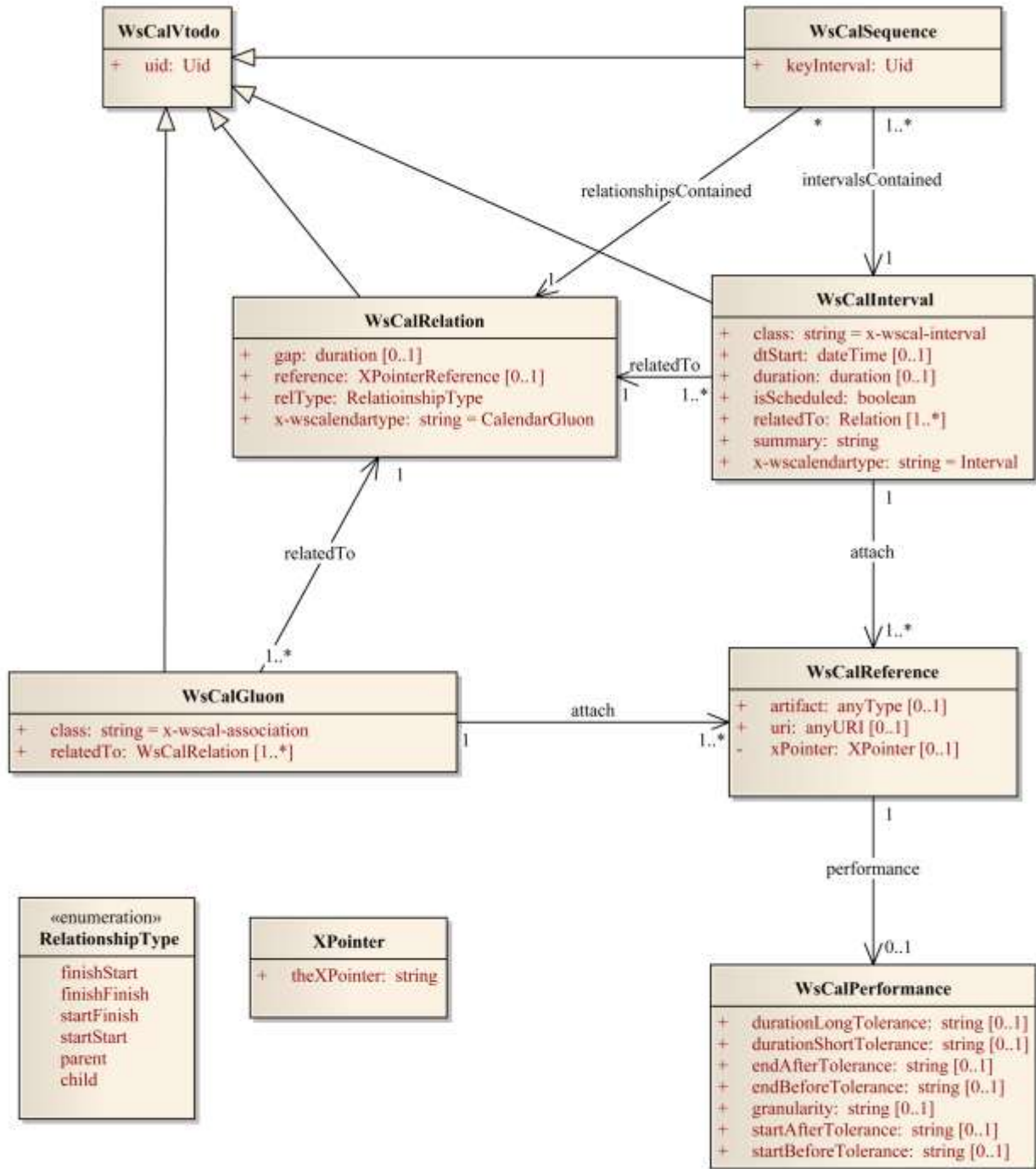


Figure 1: Abstract UML Model

1684
1685
1686
1687

1688 **6.2 Implementation Model for WS-Calendar**



1689
1690

Figure 2: Implementation Model for WS-Calendar

1691 7 Calendar Services

1692 The Service interactions are built upon and make the same assumptions about structure as the CalDAV
1693 protocol defined in **[RFC4791]** and related specifications. It does NOT require nor assume the WebDAV
1694 nor CalDAV protocol but does make use of some of the same elements and structures in the CalDAV
1695 XML namespace.

1696 Calendar resources, for example events and tasks are stored as named resources (files) inside special
1697 collections (folders) known as "**Calendar Collections**".

1698 These services can be looked upon as a layer built on top of CalDAV and defines the basic operations
1699 which allow creation, retrieval, update and deletion. In addition, query, and free-busy operations are
1700 defined to allow efficient, partial retrieval of calendar data.

1701 These services assume a degree of conformity with CalDAV is established such that services built in that
1702 manner do not have a significant mismatch. It is assumed that some WS-Calendar services will be built
1703 without any CalDAV support.

1704 7.1 Overview of the protocol

1705 The protocol is an HTTP based RESTful protocol using a limited set of methods. Each request may be
1706 followed by a response containing status information.

1707 The following methods are specified in the protocol description, PUT, POST, GET, DELETE. To avoid
1708 various issues with certain methods being blocked clients may use the X-HTTP-Method-Override: header
1709 to specify the intended operation. Servers SHOULD behave as if the named method was used.

```
1710 POST /user/fred/calendar/ HTTP/1.1  
1711 ...  
1712 X-HTTP-Method-Override: PUT  
1713 Properties
```

1714 A service or resource will have a number of properties which describe the current state of that service or
1715 resource. These properties are accessed through a GET on the target resource or service with an
1716 ACCEPT header specifying application/xrd+xml. See Section 7.1.3.6

1717 The following operations are defined by this specification:

- 1718 • Retrieval and update of service and resource properties
- 1719 • Creation of a calendar object
- 1720 • Retrieval of a calendar object
- 1721 • Update of a calendar object
- 1722 • Deletion of a calendar object
- 1723 • Query
- 1724 • Free-busy query

1725 7.1.1 Calendar Object Resources

1726 The same restrictions apply to Calendar Object Resources as specified in CalDAV **[RFC4791]** section
1727 4.2. An additional constraint for CalWS is that no timezone specifications are transferred.

1728 7.1.2 Timezone information

1729 It is assumed that the client and server each have access to a full set of up to date timezone information.
1730 Timezones will be referenced by a timezone identifier from the full set of Olson data together with a set of
1731 well-known aliases defined [where?]. CalWS services may advertise themselves as timezone servers
1732 through the server properties object.

1733 **7.1.3 Issues not addressed by this specification.**

1734 A number of issues are not addressed by this version of the specification, either because they should be
1735 addressed elsewhere or will be addressed at some later date.

1736 **7.1.3.1 Access Control**

1737 It is assumed that the targeted server will set an appropriate level of access based on authentication. This
1738 specification will not attempt to address the issues of sharing or ACLs.

1739 **7.1.3.2 Provisioning**

1740 The protocol will not provide any explicit provisioning operations. If it is possible to authenticate or
1741 address a principals calendar resources then they **MUST** be automatically created if necessary or
1742 appropriate

1743 **7.1.3.3 Copy/Move**

1744 These operations are not yet defined for this version of the CalWS protocol. Both operations raise a
1745 number of issues. In particular implementing a move operation through a series of retrievals, insertions
1746 and deletions may cause undesirable side-effects. Both these operations will be defined in a later version
1747 of this specification.

1748 **7.1.3.4 Creating Collections**

1749 We will not address the issue of creating collections within the address space. The initial set is created by
1750 provisioning.

1751 **7.1.3.5 Retrieving collections**

1752 This operation is currently undefined. A GET on a collection may fail or return a complete calendar object
1753 representing the collection.

1754 **7.1.3.6 Setting service and resource properties.**

1755 These operations are not defined in this version of the specification. In the future it will be possible to
1756 define or set the properties for the service or resources within the service.

1757 **7.1.4 CalWS Glossary**

1758 **7.1.4.1 Hrefs**

1759 An href is a URI reference to a resource, for example

1760 `"http://example.org/user/fred/calendar/event1.ics".`

1761 The URL above reflects a possible structure for a calendar server. All URLs should be absolute or path-
1762 absolute following the rules defined in **RFC4918** Section 8.3.

1763 **7.1.4.2 Calendar Object Resource**

1764 A calendar object resource is an event, meeting or a task. Attachments are resources but **NOT** calendar
1765 object resources. An event or task with overrides is a single calendar resource entity.

1766 **7.1.4.3 Calendar Collection**

1767 A folder only allowed to contain calendar object resources.

1768 7.1.4.4 Scheduling Calendar Collection

1769 A folder only allowed to contain calendar resources which is also used for scheduling operations.
1770 Scheduling events placed in such a collection will trigger implicit scheduling activity on the server.

1771 7.1.4.5 Principal Home

1772 The collection under which all the resources for a given principal are stored. For example, for principal
1773 "fred" the principal home might be "/user/fred/"

1774 7.2 Error conditions

1775 Each operation on the calendar system has a number of pre-conditions and post-conditions that apply.

1776 A "precondition" for a method describes the state of the server that must be true for that method to be
1777 performed. A "postcondition" of a method describes the state of the server that must be true after that
1778 method has been completed. Any violation of these conditions will result in an error response in the form
1779 of a CalWS XML error element containing the violated condition and an optional description. \

1780 Each method specification defines the preconditions that must be satisfied before the method can
1781 succeed. A number of postconditions are generally specified which define the state that must exist after
1782 the execution of the operation. Preconditions and postconditions are defined as error elements in the
1783 CalWS XML namespace.

1784 7.2.1 Example: error with CalDAV error condition

```
1785 <?xml version="1.0" encoding="utf-8"  
1786     xmlns:CW="Error! Reference source not found."  
1787     xmlns:C="urn:ietf:params:xml:ns:caldav" ?>  
1788 <CW:error>  
1789   <C:supported-filter>  
1790     <C:prop-filter name="X-ABC-GUID"/>  
1791   </C:supported-filter>  
1792   <CW:description>Unknown property </CW:description>  
1793 </CW:error>
```


1794

8 Properties and link relations

1795

8.1 Property and relation-type URIs

1796

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.

1797

1798

1799

1800

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

1801

1802

1803

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

1804

corresponds to

1805

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

1806

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`.

1807

1808

8.2 supported-features property.

1809

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

1810

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

1811

1812

1813

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

1814

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

1815

1816

8.3 max-attendees-per-instance

1817

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

1818

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

1819

8.4 max-date-time

1820

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

1821

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

1822

8.5 max-instances

1823

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

1824

Defines the maximum number of instances allowed per event or task

1825

8.6 max-resource-size

1826

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

1827

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.

1828

1829 **8.7 min-date-time**

1830 <http://docs.oasis-open.org/ns/wscal/calws/min-date-time>

1831 Provides a DATE-TIME value indicating the earliest date and time (in UTC) that the server is willing to
1832 accept for any DATE or DATE-TIME value in a calendar object resource stored in a calendar collection.

1833 **8.8 description**

1834 <http://docs.oasis-open.org/ns/wscal/calws/description>

1835 Provides some descriptive text for the targeted collection.

1836 **8.9 timezone-service relation.**

1837 <http://docs.oasis-open.org/ns/wscal/calws/timezone-service>

1838 The location of a timezone service used to retrieve timezone information and specifications. This may be
1839 an absolute URL referencing some other service or a relative URL if the current server also provides a
1840 timezone service.

```
1841 <Link rel="http://docs.oasis-open.org/ns/wscal/calws/calws/timezone-service"  
1842 href="http://example.com/tz" />
```

1843 **8.10 principal-home relation.**

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

1845 Provides the URL to the user home for the currently authenticated principal.

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

1848 **8.11 current-principal-freebusy relation.**

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

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

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

1853 **8.12 principal-freebusy relation.**

1854 <http://docs.oasis-open.org/ns/wscal/calws/principal-freebusy>

1855 Provides the URL to use as a target for freebusy requests for a different principal.

```
1856 <Link rel="http://docs.oasis-open.org/ns/wscal/calws/principal-freebusy"  
1857 href="http://example.com/freebusy" />
```

1858 **8.13 child-collection relation.**

1859 <http://docs.oasis-open.org/ns/wscal/calws/child-collection>

1860 Provides information about a child collections for the target. The href attribute gives the URI of the
1861 collection. The element should only have CalWS child elements giving the type of the collection, that is
1862 the CalWS:collection link property and the CalWS-calendar-collection link property. This allows clients to
1863 determine the structure of a hierarchical system by targeting each of the child collections in turn.

1864 The xrd:title child element of the link element provides a description for the child-collection.

```
1865 <Link rel="http://http://docs.oasis-open.org/ns/wscal/calws/child-collection"  
1866 href="http://example.com/calws/user/fred/calendar">  
1867 <Title xml:lang="en">Calendar</Title>  
1868 <Property type="http://docs.oasis-open.org/ns/wscal/calws/collection"  
1869 xsi:nil="true" />
```


1870
1871
1872
1873

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

1874 8.14 created link property

1875 <http://docs.oasis-open.org/ns/wscal/calws/created>

1876 Appears within a link relation describing collections or entities. The value is a date-time as defined in
1877 RFC3339 Section 5.6

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

1880 8.15 last-modified property

1881 <http://docs.oasis-open.org/ns/wscal/calws/last-modified>

1882 Appears within an xrd object describing collections or entities. The value is the same format as would
1883 appear in the Last-Modified header and is defined in **Error! Reference source not found.** Section 3.3.1

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

1886 8.16 displayname property

1887 <http://docs.oasis-open.org/ns/wscal/calws/displayname>

1888 Appears within an xrd object describing collections or entities. The value is a localized name for the entity
1889 or collection.

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

1892 8.17 timezone property

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

1894 Appears within an xrd object describing collections. The value is a text timezone identifier.

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

1897 8.18 owner property

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

1899 Appears within an xrd object describing collections or entities. The value is a server specific uri.

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

1902 8.19 collection link property

1903 <http://docs.oasis-open.org/ns/wscal/calws/collection>

1904 Appears within a link relation describing collections or entities. The property takes no value and indicates
1905 that this child element is a collection.

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

1908 8.20 calendar-collection link property

1909 <http://docs.oasis-open.org/ns/wscal/calws/calendar-collection>

1910 Appears within a link relation describing collections or entities. The property takes no value and indicates
1911 that this child element is a calendar collection.

```
1912 <Property type="http://docs.oasis-open.org/ns/wscal/calws/calendar-collection"  
1913 xsi:nil="true" />
```

1914 8.21 CalWS:privilege-set XML element

1915 <http://docs.oasis-open.org/ns/wscal/calws:privilege-set>

1916 Appears within a link relation describing collections or entities and specifies the set of privileges allowed
1917 to the current authenticated principal for that collection or entity.

```
1918 <!ELEMENT calws:privilege-set (calws:privilege*)>  
1919 <!ELEMENT calws:privilege ANY>
```

1920 Each privilege element defines a privilege or access right. The following set is currently defined

- 1921 • CalWS: Read - current principal has read access
- 1922 • CalWS: Write - current principal has write access

```
1923 <calws:privilege-set>  
1924 <calws:privilege><calws:read></calws:privilege>  
1925 <calws:privilege><calws:write></calws:privilege>  
1926 </calws:privilege-set>
```

1927

9 Retrieving Collection and Service Properties

1928 Properties, related services and locations are obtained from the service or from service resources in the
1929 form of an XRD document as defined by [XRD-1.0].

1930 Given the URL of a CalWS service a client retrieves the service XRD document through a GET on the
1931 service URL with an ACCEPT header specifying application/xrd+xml.

1932 Retrieving resource properties is identical to obtaining service properties, that is, execute a GET on the
1933 target URL with an ACCEPT header specifying application/xrd+xml.

1934 The service properties define the global limits and defaults. Any properties defined on collections within
1935 the service hierarchy override those service defaults. The service may choose to prevent such overriding
1936 of defaults and limits when appropriate.

1937 9.1 Request parameters

- 1938 • None

1939 9.2 Responses:

- 1940 • 200: OK
- 1941 • 403: Forbidden
- 1942 • 404: Not found

1943 9.3 Example - retrieving server properties:

```
1944 >>Request
1945
1946 GET / HTTP/1.1
1947 Host: example.com
1948 ACCEPT:application/xrd+xml
1949
1950 >>Response
1951 <XRD xmlns="http://docs.oasis-open.org/ns/xri/xrd-1.0"
1952     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
1953   <Expires>1970-01-01T00:00:00Z</Expires>
1954   <Subject>http://example.com/calws</Subject>
1955   <Property type="http://docs.oasis-open.org/ns/wscal/calws/created"
1956     >1970-01-01</Property>
1957
1958   <Link rel="http://docs.oasis-open.org/ns/wscal/calws/timezone-service"
1959     href="http://example.com/tz" />
1960
1961   <calWS:privilege-set>
1962   <calWS:privilege><calWS:read></calWS:privilege>
1963   </calWS:privilege-set>
1964
1965   <Link rel="http://docs.oasis-open.org/ns/wscal/calws/principal-home"
1966     type="collection"
1967     href="http://example.com/calws/user/fred">
1968   <Title xml:lang="en">Fred's calendar home</Title>
1969   </Link>
1970
1971   <Link rel="http://docs.oasis-open.org/ns/wscal/calws/child-collection"
1972     type="calendar,scheduling"
1973     href="http://example.com/calws/user/fred/calendar">
1974   <Title xml:lang="en">Calendar</Title>
```

1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985

```
</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>
```

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
2022
2023
2024
2025
2026

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 **Error! Reference source not found.** (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;

- 2027 • **CalWS:after-max-date-time:** The resource submitted in the PUT request, or targeted by a COPY
2028 or MOVE request, MUST have all of its iCalendar DATE or DATE-TIME property values (for each
2029 recurring instance) less than the value of the CalDAV:max-date-time property value on the calendar
2030 collection where the resource will be stored;
- 2031 • **CalWS:too-many-instances:** The resource submitted in the PUT request, or targeted by a COPY
2032 or MOVE request, MUST generate a number of recurring instances less than or equal to the value
2033 of the CalDAV: max-instances property value on the calendar collection where the resource will be
2034 stored;
- 2035 • **CalWS:too-many-attendees-per-instance:** The resource submitted in the PUT request, or
2036 targeted by a COPY or MOVE request, MUST have a number of ATTENDEE properties on any one
2037 instance less than or equal to the value of the CalDAV:max-attendees-per-instance property value
2038 on the calendar collection where the resource will be stored;

2039 10.4 Example - successful POST:

```

2040 >>Request
2041
2042 POST /user/fred/calendar/?action=create HTTP/1.1
2043 Host: example.com
2044 Content-Type: application/xml+calendar; charset="utf-8"
2045 Content-Length: ?
2046
2047 <?xml version="1.0" encoding="utf-8" ?>
2048 <icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
2049   <vcalendar>
2050     ...
2051   </vcalendar>
2052 </icalendar>
2053
2054 >>Response
2055
2056 HTTP/1.1 201 Created
2057 Location: http://example.com/user/fred/calendar/event1.ics

```

2058 10.5 Example - unsuccessful POST:

```

2059 >>Request
2060
2061 POST /user/fred/readcalendar/?action=create HTTP/1.1
2062 Host: example.com
2063 Content-Type: text/text; charset="utf-8"
2064 Content-Length: ?
2065
2066 This is not an xml calendar object
2067
2068 >>Response
2069
2070 HTTP/1.1 403 Forbidden
2071 <?xml version="1.0" encoding="utf-8"
2072   xmlns:D="DAV:"
2073   xmlns:C="urn:ietf:params:xml:ns:caldav" ?>
2074 <D:error>
2075   <C:supported-calendar-data/>
2076   <D:description>Not an icalendar object</C:description>
2077 </D:error>

```

2078

11 Retrieving resources

2079 A simple GET on the href will return a named resource. If that resource is a recurring event or task with
2080 overrides, the entire set will be returned. The desired format is specified in the ACCEPT header. The
2081 default form is application/xml+calendar

2082 11.1 Request parameters

- 2083 • none

2084 11.2 Responses:

- 2085 • 200: OK
- 2086 • 403: Forbidden - no access
- 2087 • 406 The requested format specified in the accept header is not supported.

2088 11.3 Example - successful fetch:

```
2089 >>Request
2090 GET /user/fred/calendar/event1.ics HTTP/1.1
2091 Host: example.com
2092
2093 >>Response
2094 HTTP/1.1 200 OK
2095 Content-Type: application/xml+calendar; charset="utf-8"
2096 Content-Length: ?
2097
2098 <?xml version="1.0" encoding="utf-8" ?>
2099 <icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
2100   <vcalendar>
2101     ...
2102   </vcalendar>
2103 </icalendar>
```

2106 11.4 Example - unsuccessful fetch:

```
2107 >>Request
2108 PUT /user/fred/calendar/noevent1.ics HTTP/1.1
2109 Host: example.com
2110
2111 >>Response
2112 HTTP/1.1 404 Not found
2113
2114
```

2115

12 Updating resources

2116 Resources are updated with the PUT method targeted at the resource href. The body of the request
2117 contains a complete new resource which effectively replaces the targeted resource. To allow for optimistic
2118 locking of the resource use the if-match header.

2119 When updating a recurring event all overrides and master must be supplied as part of the content.

2120 Preconditions as specified in Section 10.3 are applicable.

2121 12.1 Responses:

2122 • 200: OK

2123 • 304: Not modified - entity was modified by some other request

2124 • 403: Forbidden - no access, does not exist etc. See error response

2125

2126

Example 16: Successful Update

```
2127 >>Request
2128
2129 PUT /user/fred/calendar/event1.ics HTTP/1.1
2130 Host: example.com
2131 Content-Type: application/xml+calendar; charset="utf-8"
2132 Content-Length: ?
2133
2134 <?xml version="1.0" encoding="utf-8" ?>
2135 <icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
2136   <vcalendar>
2137     ...
2138   </vcalendar>
2139 </icalendar>
2140
2141 >>Response
2142
2143 HTTP/1.1 200 OK
```

2144

Example 17: Unsuccessful Update

```
2145 >>Request
2146
2147 PUT /user/fred/readcalendar/event1.ics HTTP/1.1
2148 Host: example.com
2149 Content-Type: application/xml+calendar; charset="utf-8"
2150 Content-Length: ?
2151
2152 <?xml version="1.0" encoding="utf-8" ?>
2153 <icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
2154   <vcalendar>
2155     ...
2156   </vcalendar>
2157 </icalendar>
2158
2159 >>Response
2160
2161 HTTP/1.1 403 Forbidden
2162 Content-Type: application/xml; charset="utf-8"
2163 Content-Length: xxxx
2164
2165 <?xml version="1.0" encoding="utf-8"
```



```
2166     xmlns:D="DAV:"
2167     xmlns:CW="Error! Reference source not found." ?>
2168 <CW:error>
2169   <CW:target-exists/>
2170   <CW:description>Target of update must exist</C:description>
2171 </CW:error>
```

2172

13 Deletion of resources

2173 Delete is defined in [RFC 2616] Section 9.7. In addition to conditions defined in that specification, servers
2174 must remove any references from the deleted resource to other resources. Resources are deleted with
2175 the DELETE method targeted at the resource URL. After a successful completion of a deletion a GET on
2176 that URL must result in a 404 - Not Found status.

2177

13.1 Delete for Collections

2178 Delete for collections may or may not be supported by the server. Certain collections are considered
2179 undeletable. On a successful deletion of a collection all contained resources to any depth must also be
2180 deleted.

2181

13.2 Responses:

2182

- 200: OK

2183

- 403: Forbidden - no access

2184

- 404: Not Found

2185 14 Querying calendar resources

2186 Querying provides a mechanism by which information can be obtained from the service through possibly
2187 complex queries. A list of icalendar properties can be specified to limit the amount of information returned
2188 to the client. A query takes the parts

- 2189 • Limitations on the data returned
- 2190 • Selection of the data
- 2191 • Optional timezone id for floating time calculations.

2192 The current specification uses CalDAV multiget and calendar-query XML bodies as specified in [RFC
2193 4791] with certain limitations and differences.

- 2194 1. The POST method is used for all requests, the action being identified by the outer element.
- 2195 2. While CalDAV servers generally only support [RFC 5545] and assume that as the default, the
2196 delivery format for CalWS will, by default, be [draft-xcal].
- 2197 3. The CalDAV query allows the specification of a number of DAV properties. Specification of these
2198 properties, with the exception of DAV:getetag, is considered an error in CalWS.
- 2199 4. The CalDAV:propnames element is invalid

2200 With those differences, the CalDAV specification is the normative reference for this operation.

2201 14.1 Limiting data returned

2202 This is achieved by specifying one of the following

- 2203 • CalDAV:allprop return all properties (some properties are specified as not being part of the allprop
2204 set so are not returned)
- 2205 • CalDAV:prop An element which contains a list of properties to be returned . May only contain
2206 DAV:getetag and CalDAV:calendar-data

2207 Of particular interest, and complexity, is the calendar-data property which can contain a time range to limit
2208 the range of recurrences returned and/or a list of calendar properties to return.

2209 14.2 Pre/postconditions for calendar queries

2210 The preconditions as defined in in [RFC 4791] Section 7.8 apply here. CalDav errors may be reported by
2211 the service when preconditions or postconditions are violated.

2212 14.3 Example: time range limited retrieval

2213 This example shows the time-range limited retrieval from a calendar which results in 2 events, one a
2214 recurring event and one a simple non-recurring event.

```
2215 >> Request <<
2216
2217 POST /user/fred/calendar/ HTTP/1.1
2218 Host: calws.example.com
2219 Depth: 1
2220 Content-Type: application/xml; charset="utf-8"
2221 Content-Length: xxxx
2222
2223 <?xml version="1.0" encoding="utf-8" ?>
2224 <C:calendar-query xmlns:D="DAV:"
2225                  xmlns:C="urn:ietf:params:xml:ns:caldav">
2226   <D:prop>
2227     <D:getetag/>
2228     <C:calendar-data content-type="application/xml+calendar" >
```

```

2229     <C:comp name="VCALENDAR">
2230         <C:prop name="VERSION"/>
2231         <C:comp name="VEVENT">
2232             <C:prop name="SUMMARY"/>
2233             <C:prop name="UID"/>
2234             <C:prop name="DTSTART"/>
2235             <C:prop name="DTEND"/>
2236             <C:prop name="DURATION"/>
2237             <C:prop name="RRULE"/>
2238             <C:prop name="RDATE"/>
2239             <C:prop name="EXRULE"/>
2240             <C:prop name="EXDATE"/>
2241             <C:prop name="RECURRENCE-ID"/>
2242         </C:comp>
2243     </C:comp>
2244 </C:calendar-data>
2245 </D:prop>
2246 <C:filter>
2247     <C:comp-filter name="VCALENDAR">
2248         <C:comp-filter name="VEVENT">
2249             <C:time-range start="20060104T000000Z"
2250                 end="20060105T000000Z"/>
2251         </C:comp-filter>
2252     </C:comp-filter>
2253 </C:filter>
2254 </C:calendar-query>
2255
2256 >> Response <<
2257
2258 HTTP/1.1 207 Multi-Status
2259 Date: Sat, 11 Nov 2006 09:32:12 GMT
2260 Content-Type: application/xml; charset="utf-8"
2261 Content-Length: xxxx
2262
2263 <?xml version="1.0" encoding="utf-8" ?>
2264 <D:multistatus xmlns:D="DAV:"
2265     xmlns:C="urn:ietf:params:xml:ns:caldav">
2266     <D:response>
2267         <D:href>http://cal.example.com/bernard/work/abcd2.ics</D:href>
2268         <D:propstat>
2269             <D:prop>
2270                 <D:getetag>"fffff-abcd2"</D:getetag>
2271                 <C:calendar-data content-type="application/xml+calendar" >
2272                     <xc:icalendar
2273                         xmlns:xc="urn:ietf:params:xml:ns:icalendar-2.0">
2274 <xc:vcalendar>
2275     <xc:properties>
2276         <xc:calscale><text>GREGORIAN</text></xc:calscale>
2277         <xc:prodid>
2278             <xc:text>-//Example Inc.//Example Calendar//EN</xc:text>
2279         </xc:prodid>
2280         <xc:version><xc:text>2.0</xc:text></xc:version>
2281     </xc:properties>
2282     <xc:components>
2283         <xc:vevent>
2284             <xc:properties>
2285                 <xc:dtstart>
2286                     <xc:parameters>
2287                         <xc:tzid>US/Eastern<xc:tzid>
2288                     </xc:parameters>
2289                     <xc:parameters>
2290                         <xc:date-time>20060102T120000</xc:date-time>
2291                     </xc:dtstart>
2292                 <xc:duration><xc:duration>PT1H</xc:duration></xc:duration>
2293             </xc:properties>
2294             <xc:summary>

```

```

2293     <xc:text>Event #2</xc:text>
2294 </xc:summary>
2295 <xc:uid>
2296     <xc:text>00959BC664CA650E933C892C@example.com</xc:text>
2297 </xc:uid>
2298 <xc:rrule>
2299     <xc:recur>
2300         <xc:freq>DAILY</xc:freq>
2301         <xc:count>5</xc:count>
2302     </xc:recur>
2303 </xc:rrule>
2304 </xc:properties>
2305 </xc:vevent>
2306
2307 <xc:vevent>
2308     <xc:properties>
2309         <xc:dtstart>
2310             <xc:parameters>
2311                 <xc:tzid>US/Eastern<xc:tzid>
2312             </xc:parameters>
2313             <xc:date-time>20060104T140000</xc:date-time>
2314         </xc:dtstart>
2315         <xc:duration><xc:duration>PT1H</xc:duration></xc:duration>
2316     </xc:properties>
2317     <xc:summary>
2318         <xc:text>Event #2 bis</xc:text>
2319     </xc:summary>
2320     <xc:uid>
2321         <xc:text>00959BC664CA650E933C892C@example.com</xc:text>
2322     </xc:uid>
2323     <xc:recurrence-id>
2324         <xc:parameters>
2325             <xc:tzid>US/Eastern<xc:tzid>
2326         </xc:parameters>
2327         <xc:date-time>20060104T120000</xc:date-time>
2328     </xc:recurrence-id>
2329     <xc:rrule>
2330         <xc:recur>
2331             <xc:freq>DAILY</xc:freq>
2332             <xc:count>5</xc:count>
2333         </xc:recur>
2334     </xc:rrule>
2335 </xc:properties>
2336 </xc:vevent>
2337
2338 <xc:vevent>
2339     <xc:properties>
2340         <xc:dtstart>
2341             <xc:parameters>
2342                 <xc:tzid>US/Eastern<xc:tzid>
2343             </xc:parameters>
2344             <xc:date-time>20060106T140000</xc:date-time>
2345         </xc:dtstart>
2346         <xc:duration><xc:duration>PT1H</xc:duration></xc:duration>
2347     </xc:properties>
2348     <xc:summary>
2349         <xc:text>Event #2 bis bis</xc:text>
2350     </xc:summary>
2351     <xc:uid>
2352         <xc:text>00959BC664CA650E933C892C@example.com</xc:text>
2353     </xc:uid>
2354     <xc:recurrence-id>
2355         <xc:parameters>
2356             <xc:tzid>US/Eastern<xc:tzid>
2357         </xc:parameters>
2358         <xc:date-time>20060106T120000</xc:date-time>

```

```

2357     </xc:recurrence-id>
2358     <xc:rrule>
2359         <xc:recur>
2360             <xc:freq>DAILY</xc:freq>
2361             <xc:count>5</xc:count>
2362         </xc:recur>
2363     </xc:rrule>
2364 </xc:properties>
2365 </xc:vevent>
2366 </xc:components>
2367 </xc:vcalendar>
2368 </xc:icalendar>
2369     </C:calendar-data>
2370 </D:prop>
2371     <D:status>HTTP/1.1 200 OK</D:status>
2372 </D:propstat>
2373 </D:response>
2374 <D:response>
2375     <D:href>http://cal.example.com/bernard/work/abcd3.ics</D:href>
2376     <D:propstat>
2377         <D:prop>
2378             <D:getetag>"fffff-abcd3"</D:getetag>
2379             <C:calendar-data content-type="application/xml+calendar" >
2380                 <xcal:icalendar
2381                     xmlns:xc="urn:ietf:params:xml:ns:icalendar-2.0">
2382 <xc:vcalendar>
2383 <xc:properties>
2384 <xc:calscale><text>GREGORIAN</text></xc:calscale>
2385 <xc:prodid>
2386 <xc:text>-//Example Inc.//Example Calendar//EN</xc:text>
2387 </xc:prodid>
2388 <xc:version><xc:text>2.0</xc:text></xc:version>
2389 </xc:properties>
2390 <xc:components>
2391 <xc:vevent>
2392 <xc:properties>
2393 <xc:dtstart>
2394 <xc:parameters>
2395 <xc:tzid>US/Eastern<xc:tzid>
2396 <xc:parameters>
2397 <xc:date-time>20060104T100000</xc:date-time>
2398 </xc:dtstart>
2399 <xc:duration><xc:duration>PT1H</xc:duration></xc:duration>
2400 <xc:summary>
2401 <xc:text>Event #3</xc:text>
2402 </xc:summary>
2403 <xc:uid>
2404 <xc:text>DC6C50A017428C5216A2F1CD@example.com</xc:text>
2405 </xc:uid>
2406 <xc:rrule>
2407 <xc:recur>
2408 <xc:freq>DAILY</xc:freq>
2409 <xc:count>5</xc:count>
2410 </xc:recur>
2411 </xc:rrule>
2412 </xc:properties>
2413 </xc:vevent>
2414 </xc:components>
2415 </xc:vcalendar>
2416 </xc:icalendar>
2417 </C:calendar-data>
2418 </D:prop>
2419 <D:status>HTTP/1.1 200 OK</D:status>
2420 </D:propstat>

```

2421
2422

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

2423

15 Free-busy queries

2424 Free-busy queries are used to obtain free-busy information for a calendar-collection or principals. The
2425 result contains information only for events to which the current principal has sufficient access.

2426 When targeted at a calendar collection the result is based only on the calendaring entities contained in
2427 that collection. When targeted at a principal free-busy URL the result will be based on all information
2428 which affect the principals free-busy status, for example availability.

2429 The possible targets are:

- 2430 • A calendar collection URL
- 2431 • The XRD link with relation CalWS/current-principal-freebusy
- 2432 • The XRD link with relation CalWS/principal-freebusy with a principal given in the request.

2433 The query follows the specification defined in Error! Reference source not found. with certain limitations.
2434 As an authenticated user to the CalWS service scheduling read-freebusy privileges must have been
2435 granted. As an unauthenticated user equivalent access must have been granted to unauthenticated
2436 access.

2437 Freebusy information is returned by default as xcalendar vfreebusy components, as defined by **[draft-
2438 xcal]**. Such a component is not meant to conform to the requirements of VFREEBUSY components in
2439 **[RFC 5546]**. The VFREEBUSY component SHOULD conform to section "4.6.4 Free/Busy Component" of
2440 **[RFC 5545]**. A client SHOULD ignore the ORGANIZER field..

2441 Since a Freebusy query can only refer to a single user, a client will already know how to match the result
2442 component to a user. A server MUST only return a single vfreebusy component.

2443 15.1 ACCEPT header

2444 The Accept header is used to specify the format for the returned data. In the absence of a header the
2445 data should be returned as specified in **[draft-xcal]**, that is, as if the following had been specified

2446 `ACCEPT: application/xml+calendar`

2447 15.2 URL Query Parameters

2448 None of these parameters are required except for the conditions noted below. Appropriate defaults will be
2449 supplied by the server.

2450 15.2.1 start

2451 **Default:** The default value is left up to the server. It may be the current day, start of the current
2452 month, etc.

2453 **Description:** Specifies the start date for the Freebusy data. The server is free to ignore this value and
2454 return data in any time range. The client must check the data for the returned time range.

2455 **Format:** A profile of an **[RFC3339]** Date/Time. Fractional time is not supported. The server MUST
2456 support the expanded version e.g.

2457 `2007-01-02T13:00:00-08:00`

2458 It is up to the server to interpret local date/times.

2459 **Example:**

2460 `2007-02-03T15:30:00-0800`

2461 `2007-12-01T10:15:00Z`

2462 **Notes:** Specifying only a start date/time without specifying an end-date/time or period should be
2463 interpreted as in **[RFC 5545]**. The effective period should cover the remainder of that day.

2464 Date-only values are disallowed as the server cannot determine the correct start of the day. Only

2465 UTC or date/time with offset values are permitted.

2466 15.2.2 end

2467 **Default:** Same as start

2468 **Description:** Specifies the end date for the Freebusy data. The server is free to ignore this value.

2469 **Format:** Same as start

2470 **Example:** Same as start

2471 15.2.3 period

2472 **Default:** The default value is left up to the server. The recommended value is "P42D".

2473 **Description:** Specifies the amount of Freebusy data to return. A client cannot specify both a period
2474 and an end date. Period is relative to the start parameter.

2475 **Format:** A duration as defined in section 4.3.6 of [RFC 5545]

2476 **Example:**

2477 `P42D`

2478 15.2.4 account

2479 **Default:** none

2480 **Description:** Specifies the principal when the request is targeted at the XRD CalWS/principal-
2481 freebusy. Specification of this parameter is an error otherwise.

2482 **Format:** Server specific

2483 **Example:**

2484 `fred`
2485 `/principals/users/jim`
2486 `user1@example.com`

2487 15.3 URL parameters - notes

2488 The server is free to ignore the start, end and period parameters. It is recommended that the server return
2489 at least 6 weeks of data from the current day.

2490 A client MUST check the time range in the VFREEBUSY response as a server may return a different time
2491 range than the requested range.

2492 15.4 HTTP Operations

2493 The server SHOULD return an Etag response header for a successful GET request targeting a Freebusy
2494 read URL. Clients MAY use the Etag response header value to do subsequent "conditional" GET
2495 requests that will avoid re-sending the Freebusy data again if it has not changed.

2496 15.5 Response Codes

2497 Below are the typical status codes returned by a GET request targeting a Free-busy URL. Note that other
2498 HTTP status codes not listed here might also be returned by a server.

- 2499
- 200 OK
- 2500
- 302 Redirect
- 2501
- 400 Start parameter could not be understood / End parameter could not be understood / Period
- 2502
- parameter could not be understood
- 2503
- 401 Unauthorized

- 2504 • 403 Forbidden
- 2505 • 404 The data for the requested principal is not currently available, but may be available later.
- 2506 • 406 The requested format in the accept header is not supported.
- 2507 • 410 The data for the requested principal is no longer available
- 2508 • 500 General server error

2509 15.6 Examples

2510 The following are examples of URLs used to retrieve Free-busy data for a user:

```

2511 http://www.example.com/freebusy/user1@example.com?
2512 start=2007-09-01T00:00:00-08:00&end=2007-09-31T00:00:00-08:00
2513
2514 http://www.example.com/freebusy/user1@example.com?
2515 start=2007-09-01T00:00:00-08:00&end=2007-09-31T00:00:00-08:00
2516
2517 http://www.example.com/freebusy/user1@example.com
2518
2519 http://www.example.com/freebusy?user=user%201@example.com&
2520 start=2008-01-01T00:00:00Z&end=2008-12-31T00:00:00Z

```

2521 Some Request/Response Examples:

2522 A URL with no query parameters:

```

2523 >> Request <<
2524 GET /freebusy/bernard/ HTTP/1.1
2525 Host: www.example.com
2526
2527 >> Response <<
2528 HTTP/1.1 200 OK
2529 Content-Type: application/xml+calendar; charset="utf-8"
2530 Content-Length: xxxx
2531
2532 <xc:icalendar xmlns:xc="urn:ietf:params:xml:ns:icalendar-2.0">
2533   <xc:vcalendar>
2534     <xc:properties>
2535       <xc:calscale><text>GREGORIAN</text></xc:calscale>
2536       <xc:prodid>
2537         <xc:text>-//Example Inc.//Example Calendar//EN</xc:text>
2538       </xc:prodid>
2539       <xc:version><xc:text>2.0</xc:text></xc:version>
2540     </xc:properties>
2541     <xc:components>
2542       <xc:vfreebusy>
2543         <xc:properties>
2544           <xc:uid>
2545             <xc:text>76ef34-54a3d2@example.com</xc:text>
2546           </xc:uid>
2547           <xc:dtstart>
2548             <xc:date-time>20060101T000000Z</xc:date-time>
2549           </xc:dtstart>
2550           <xc:dtend>
2551             <xc:date-time>20060108T000000Z</xc:date-time>
2552           </xc:dtend>
2553           <xc:dtstamp>
2554             <xc:date-time>20050530T123421Z</xc:date-time>
2555           </xc:dtstamp>
2556           <xc:freebusy>
2557             <xc:parameters>
2558               <xc:fbtype>BUSYTENTATIVE<xc:fbtype>
2559             <xc:parameters>
2560               <xc:period>20060102T100000Z/20060102T120000Z</xc:period>

```

```

2561     </xc:freebusy>
2562     <xc:freebusy>
2563         <xc:period>20060103T100000Z/20060103T120000Z</xc:period>
2564     </xc:freebusy>
2565     <xc:freebusy>
2566         <xc:period>20060104T100000Z/20060104T120000Z</xc:period>
2567     </xc:freebusy>
2568     <xc:freebusy>
2569         <xc:parameters>
2570             <xc:fbtype>BUSYUNAVAILABLE<xc:fbtype>
2571         <xc:parameters>
2572         <xc:period>20060105T100000Z/20060105T120000Z</xc:period>
2573     </xc:freebusy>
2574     <xc:freebusy>
2575         <xc:period>20060106T100000Z/20060106T120000Z</xc:period>
2576     </xc:freebusy>
2577 </xc:vfreebusy>
2578 </xc:components>
2579 </xc:vcalendar>
2580 <xc:icalendar>

```

2581 A URL with start and end parameters:

```

2582 >> Request <<
2583 GET /freebusy/user1@example.com?start=2007-09-01T00:00:00-08:00&end=2007-09-
2584 31T00:00:00-08:00
2585 HTTP/1.1
2586 Host: www.example.com
2587
2588 >> Response <<
2589 HTTP/1.1 200 OK
2590 Content-Type: application/xml+calendar; charset="utf-8"
2591 Content-Length: xxxx
2592
2593 <xc:icalendar xmlns:xc="urn:ietf:params:xml:ns:icalendar-2.0">
2594     <xc:vcalendar>
2595         <xc:properties>
2596             <xc:calscale><text>GREGORIAN</text></xc:calscale>
2597             <xc:prodid>
2598                 <xc:text>-//Example Inc.//Example Calendar//EN</xc:text>
2599             </xc:prodid>
2600             <xc:version><xc:text>2.0</xc:text></xc:version>
2601         </xc:properties>
2602         <xc:components>
2603             <xc:vfreebusy>
2604                 <xc:properties>
2605                     <xc:uid>
2606                         <xc:text>76ef34-54a3d2@example.com</xc:text>
2607                     </xc:uid>
2608                     <xc:dtstart>
2609                         <xc:date-time>20070901T000000Z</xc:date-time>
2610                     </xc:dtstart>
2611                     <xc:dtend>
2612                         <xc:date-time>20070931T000000Z</xc:date-time>
2613                     </xc:dtend>
2614                     <xc:dtstamp>
2615                         <xc:date-time>20050530T123421Z</xc:date-time>
2616                     </xc:dtstamp>
2617                     <xc:freebusy>
2618                         <xc:period>20070915T230000Z/20070916T010000Z</xc:period>
2619                     </xc:freebusy>
2620                 </xc:vfreebusy>
2621             </xc:components>
2622         </xc:vcalendar>
2623     </xc:icalendar>

```

2624 A URL for which the server does not have any data for that user:

```
2625 >> Request <<  
2626 GET /freebusy/user1@example.com?start=2012-12-01T00:00:00-08:00&end=2012-12-  
2627 31T00:00:00-08:00  
2628 HTTP/1.1  
2629 Host: www.example.com  
2630  
2631 >> Response <<  
2632 HTTP/1.1 404 No data  
2633
```

2634 **16 Conformance**

2635 WS-Calendar Intervals SHALL have a Duration. Intervals MAY have a StartTime. Intervals SHALL NOT
2636 include an END time. If a non-compliant Interval is received with an END time, it may be ignored.

2637 A performance component SHALL not include Start, Stop, and Duration elements. Two out of the three
2638 elements is acceptable, but not three.

2639 In Partitions, the Description, Summary and Priority of each Interval SHALL be excluded.

2640 An Calendar Gluon may either have a dtStart or a dtEnd, but may not have both.

2641 *All OASIS specifications require conformance*

2642

A. Acknowledgements

2643 The following individuals have participated in the creation of this specification and are gratefully
2644 acknowledged:

2645 **Participants:**

2646 Bruce Bartell, Southern California Edison
2647 Brad Benson, Trane
2648 Edward Cazalet, Individual
2649 Toby Considine, University of North Carolina at Chapel Hill
2650 William Cox, Individual
2651 Sharon Dinges, Trane
2652 Craig Gemmill, Tridium, Inc.
2653 Girish Ghatikar, Lawrence Berkeley National Laboratory
2654 Gerald Gray, Southern California Edison
2655 Gale Horst, Electric Power Research Institute (EPRI)
2656 Gershon Janssen, Individual
2657 Ed Koch, Akuacom Inc.
2658 Benoit Lepeuple, LonMark International*
2659 Carl Mattocks, CheckMi*
2660 Robert Old, Siemens AG
2661 Alexander Papaspyrou, Technische Universitat Dortmund
2662 Jeremy Roberts, LonMark International*
2663 David Thewlis, CalConnect

2664

2665

2666 The Calendaring and Scheduling Consortium (CalConnect) TC-XML committee worked closely with WS-
2667 Calendar Technical Committee, bridging to developing IETF standards and contributing the Services
2668 definitions that make up Services beginning in Section 7, Calendar Services. The Technical Committee
2669 gratefully acknowledges their assistance and cooperation as well. Contributors to TC XML include:

2670 Cyrus Daboo, Apple
2671 Mike Douglas, Rensselaer Polytechnic Institute
2672 Steven Lees, Microsoft
2673 Tong Li, IBM

2674

2675

B. An Introduction to Internet Calendaring

2676 *The WS-Calendar Technical Committee thanks CalConnect for contributing this overview of iCalendar*
2677 *and its use.*

2678 B.1 icalendar

2679 B.1.1 History

2680 The iCalendar specification was first produced by the IETF in 1998 as RFC 2445 [1]. Since then it has
2681 become the dominant standard for calendar data interchange on the internet and between devices
2682 (desktop computers, mobile phones etc). The specification was revised in 2009 as RFC 5545 [4].

2683 Alongside iCalendar is the iTIP specification (RFC 2446 [2] and revised as RFC 5546[5]) that defines how
2684 iCalendar is used to carry out scheduling operations (for example, how an organizer can invite attendees
2685 to a meeting and receive their replies). This forms the basis for email-based scheduling using iMIP (the
2686 specification that describes how to use iTIP with email - RFC 2447 [3]).

2687 iCalendar itself is a text-based data format. However, an XML format is also available, providing a one-to-
2688 one mapping to the text format (draft [7]).

2689 iCalendar data files typically have a .ics file name extension. Most desktop calendar clients can import or
2690 export iCalendar data, or directly access such data over the Internet using a variety of protocols.

2691 B.1.2 Data model

2692 The iCalendar data format has a well defined data model. "iCalendar objects" encompass a set of
2693 "iCalendar components" each of which contains a set of "iCalendar properties" and possibly other sub-
2694 components. An iCalendar property consists of a name, a set of optional parameters (specified as "key-
2695 value" pairs) and a value.

2696 iCalendar components include:

2697 "VEVENT" which represents an event

2698 "VTODO" which represents a task or to-do

2699 "VJOURNAL" which represents a journal entry

2700 "VFREEBUSY" which represents periods of free or busy time information

2701 "VTIMEZONE" which represents a timezone definition (timezone offset and daylight saving rules)

2702 "VALARM" is currently the only defined sub-component and is used to set alarms or reminders on events
2703 or tasks.

2704 Properties include:

2705 "DTSTART" which represents a start time for a component

2706 "DTEND" which represents an end time for a component

2707 "SUMMARY" which represents a title or summary for a component

2708 "RRULE" which can specify rules for repeating events or tasks (for example, every day, every week on
2709 Tuesdays, etc.)

2710 "ORGANIZER" which represents the calendar user who is organizing an event or assigning a task

2711 "ATTENDEE" which represents calendar users attending an event or assigned a task

2712 In addition to this data model and the pre-defined properties, the specification defines how all those are
2713 used together to define the semantics of calendar objects and scheduling. The semantics are basically a
2714 set of rules stating how all the components and properties are used together to ensure that all iCalendar
2715 products can work together to achieve good interoperability. For example, a rule requires that all events
2716 must have one and only one "DTSTART" property. The most important part of the iCalendar specification

2717 is the semantics of the calendaring model that it represents. The use of text or XML to encode those is
2718 secondary.

2719 **B.1.3 Scheduling**

2720 The iTIP specification defines how iCalendar objects are exchanged in order to accomplish the key task
2721 needed to schedule events or tasks. An example of a simple workflow is as follows:

- 2722 1. To schedule an event, an organizer creates the iCalendar object representing the event and adds
2723 calendar users as attendees.
- 2724 2. The organizer then sends an iTIP "REQUEST" message to all the attendees.
- 2725 3. Upon receipt of the scheduling message, each attendee can decide whether they want to attend
2726 the meeting or not.
- 2727 4. Each attendee can then respond back to the organizer using an iTIP "REPLY" message
2728 indicating their own attendance status.

2729 iTIP supports other types of scheduling messages, for example, to cancel meetings, add new instances to
2730 a repeating meeting, etc.

2731 **B.1.4 Extensibility**

2732 iCalendar was designed to be extensible, allowing for new components, properties and parameters to be
2733 defined as needed. A registry exists to maintain the list of standard extensions with references to their
2734 definitions to ensure anyone can use them and work well with others.

2735 **B.2 Calendar data access and exchange protocols**

2736 **B.2.1 Internet Calendar Subscriptions**

2737 An Internet calendar subscription is simply an iCalendar data file made available on a web server. Users
2738 can use this data in two ways:

- 2739 – The data can be downloaded from the web server and then imported directly into an iCalendar
2740 aware client. This solution works well for calendar data that is not likely to change over time (for
2741 example the list of national holidays for the next year).
- 2742 – Calendar clients that support "direct" subscriptions can use the URL to the calendar data on the
2743 web server to download the calendar data themselves. Additionally, the clients can check the web
2744 server on a regular basis for updates to the calendar data, and then update their own cached
2745 copy of it. This allows calendar data that changes over time to be kept synchronized.

2746 **B.2.2 CalDAV**

2747 CalDAV is a calendar access protocol and is defined in RFC 4791 [6]. The protocol is based on WebDAV
2748 which is an extension to HTTP that provides enhanced capabilities for document management on web
2749 servers.

2750 CalDAV is used in a variety of different environments, ranging from very large internet service providers,
2751 to large and small corporations or institutions, and to small businesses and individuals.

2752 CalDAV clients include desktop applications, mobile devices and browser-based solutions. It can also be
2753 used by "applets", for example, a web page panel that displays a user's upcoming events.

2754 One of the key aspects of CalDAV is its data model. Simply put, it defines a "calendar home" for each
2755 calendar user, within which any number of "calendars" can be created. Each "calendar" can contain any
2756 number of iCalendar objects representing individual events, tasks or journal entries. This data model
2757 ensures that clients and servers can interoperate well.

2758 In addition to providing simple operations to read, write and delete calendar data, CalDAV provides a
2759 querying mechanism to allow clients to fetch calendar data matching specific criteria. This is commonly

2760 used by clients to do "time-range" queries, i.e., find the set of events that occur within a given start/end
2761 time period.

2762 CalDAV also supports access control allowing for features such as delegated calendars and calendar
2763 sharing.

2764 CalDAV also specifies how scheduling operations can be done using the protocol. Whilst it uses the
2765 semantics of the iTIP protocol, it simplifies the process by allowing simple calendar data write operations
2766 to trigger the sending of scheduling messages, and it has the server automatically process the receipt of
2767 scheduling messages. Scheduling can be done with other users on the CalDAV server or with calendar
2768 users on other systems (via some form of "gateway").

2769 **B.2.3 ActiveSync/SyncML**

2770 ActiveSync and SyncML are technologies that allow multiple devices to synchronize data with a server,
2771 with calendar data being one of the classes of data supported. These have typically been used for low-
2772 end and high-end mobile devices.

2773 **B.2.4 CalWS**

2774 CalWS is a web services calendar access API developed by The Calendaring and Scheduling
2775 Consortium and the OASIS organization, to be used as part of the Oasis WS-Calendar standard. It
2776 provides an API to access and manipulate calendar data stored on a server. It follows a similar data
2777 model to CalDAV and has been designed to co-exist with a CalDAV service offering the same data.

2778 **B.2.5 iSchedule**

2779 iSchedule is a protocol to allow scheduling between users on different calendaring systems and across
2780 different internet domains. It transports iTIP scheduling messages using HTTP between servers. Servers
2781 use DNS and various security mechanisms to determine the authenticity of messages received.

2782 It has been specifically designed to be independent of any calendar system in use at the endpoints, so
2783 that it is compatible with many different systems. This allows organizations with different calendar
2784 systems to exchange scheduling messages with each other, and also allows a single organization with
2785 multiple calendar systems (for example due to mergers, or different departmental requirements) to
2786 exchange scheduling messages between users of each system.

2787 **B.3 References**

2788 [1] <https://datatracker.ietf.org/doc/rfc2445/> : 'Internet Calendaring and Scheduling Core Object
2789 Specification'

2790 [2] <https://datatracker.ietf.org/doc/rfc2446/> : 'iCalendar Transport-Independent Interoperability Protocol'

2791 [3] <https://datatracker.ietf.org/doc/rfc2447/> : 'iCalendar Message-Based Interoperability Protocol'

2792 [4] <https://datatracker.ietf.org/doc/rfc5545/> : 'Internet Calendaring and Scheduling Core Object
2793 Specification'

2794 [5] <https://datatracker.ietf.org/doc/rfc5546/> : 'iCalendar Transport-Independent Interoperability Protocol'

2795 [6] <https://datatracker.ietf.org/doc/rfc4791/> : 'Calendaring Extensions to WebDAV'

2796 [7] <https://datatracker.ietf.org/doc/draft-daboo-et-al-icalendar-in-xml/> : 'xCal: The XML format for
2797 iCalendar'

2798

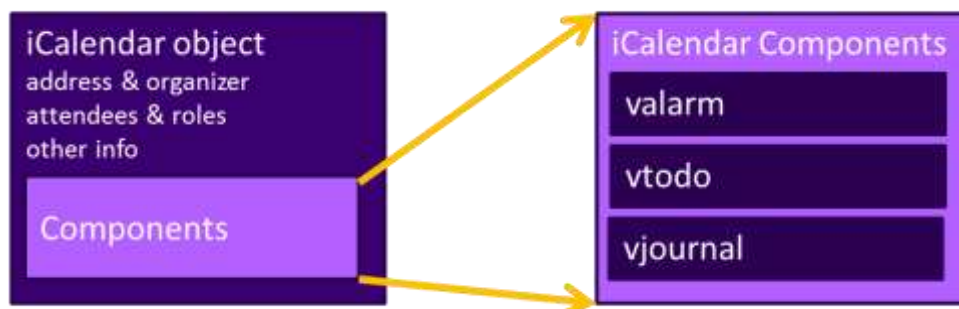
2799
2800

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

2801 iCalendar has long been the predominant message format for an Internet user to send meeting requests
2802 and tasks to other Internet users by email. The recipient can respond to the sender easily or counter
2803 propose another meeting date/time. iCalendar support is built into all major email systems and email
2804 clients. While SMTP is the predominant means to transport iCalendar messages, protocols including
2805 WebDAV and SyncML are used to transport collections of iCalendar information. No similar standard for
2806 service interactions has achieved similar widespread use.

2807 The Calendar and Scheduling Consortium (CalConnect), working within the IETF, updated the iCalendar
2808 standard in the summer of 2009 to support extension ([RFC5545]). In 2010, the same group defined
2809 [XCAL], a canonical XML serialization for iCalendar, currently (08/21/2008) on the recommended
2810 standards track within the IETF. This specification supports extensions, including handling non-standard,
2811 i.e., non-iCalendar, data during message storage and retrieval.

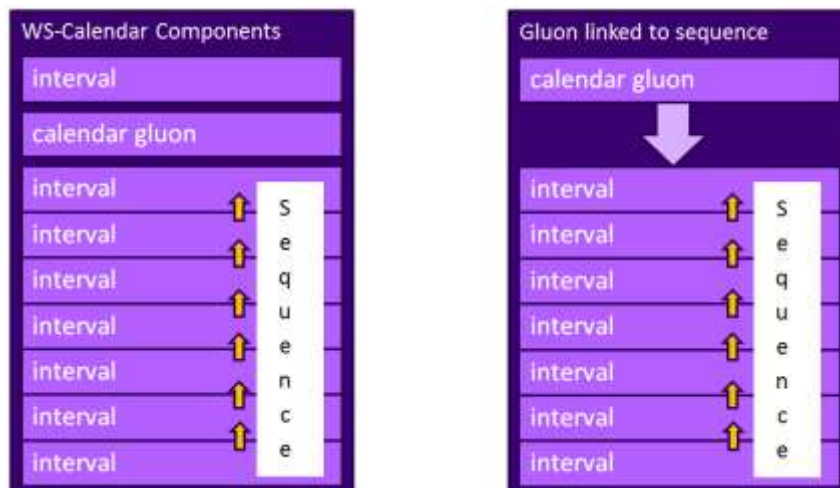
2812 WS-Calendar builds on this work, and consists of extensions to the vocabulary of iCalendar, along with
2813 standard services to extend calendaring and scheduling into service interactions. iCalendar consists of a
2814 number of fields that support the delivery, update, and synchronization of if calendar messages and a list
2815 of components. The components can specify defined relationships between each other.



2816
2817

Figure 3: iCalendar overview

2818 WS-Calendar defines the Interval, a profile of the vtodo component requiring only a duration and an
2819 artifact to define service delivery and performance. WS-Calendar also defines the CalendarGluon
2820 component, a container for holding only a service delivery and performance artifact, to associate with a
2821 component or group of components.



2822

2823

Figure 4: WS-Calendar and EMIX

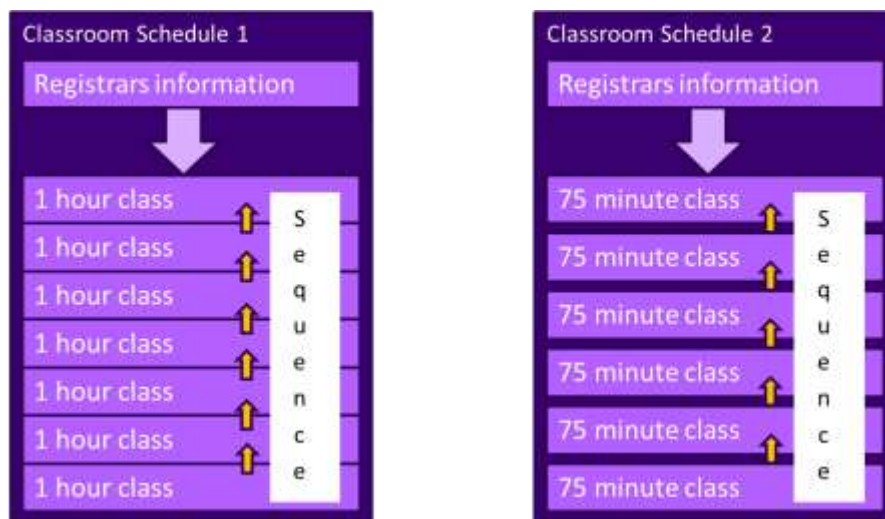
2824 A set of intervals that have defined temporal relationships is a Sequence. Temporal relationships express
2825 how the occurrence of one interval is related to another. For example, Interval B may begin 10 minutes
2826 after Interval A completes, or Interval D may start 5 minutes after Interval C starts. An Calendar Gluon
2827 linked to a Sequence defines service performance for all Intervals in the Sequence. Because each
2828 interval has its own service performance contract, specifications built on WS-Calendar can define rules for
2829 inheritance and over-rides with a sequence.

2830 The Partition is a sub-class of a Sequence in which all Intervals follow consecutively with no lag time.
2831 Intervals in a Partition normally have the same Duration, but WS-Calendar does support overriding the
2832 duration on an individual basis.

2833 C.1 Scheduling Sequences

2834 A Sequence is a general pattern of behaviors and results that does not require a specific schedule. A
2835 publishing service may advertise a Sequence with no schedule, i.e., no specific time for performance.
2836 When the Sequence is invoked or contracted, a specific performance time is added. In the original
2837 iCalendar components, this would add the starting date and time (dtStart) to the component. In WS-
2838 Calendar, we add the starting date and time only to the first Interval of a Sequence; the performance
2839 times for all other Intervals in the Sequence are derived from that one start time.

2840 C.1.1 Academic Scheduling example



2841

2842

Figure 5: Classroom Scheduling Example

2843 A college campus uses two schedules to schedule its buildings. In Schedule 1, classes start on the hour,
2844 and follow one after another; each class starts on the hour. In the second schedule, each class lasts an
2845 hour and a quarter, and there is a fifteen minute gap between classes; classes start on the half hour. On
2846 many campuses, the sequence in Schedule 1 may describe classes taught on Monday, Wednesday, and
2847 Friday. Schedule 2 may describe classes taught on Tuesday and Thursday.

2848 The registrar's office knows some key facts about each classroom, including whether it hosts a class
2849 during a particular period, and the number of students that will be in that class. The college wishes to
2850 optimize the provision of building services for each class. Such services may include adequate ventilation
2851 and comfortable temperatures to assure alert students. Other services may ensure that the classroom
2852 projection systems and A/V support services are warmed up in advance of a class, or powered off when a
2853 classroom is vacant.

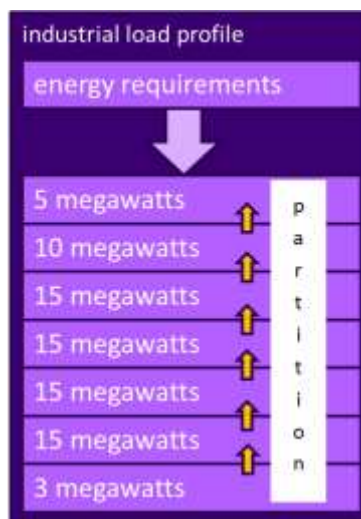
2854 Although most classes meet over typical schedule for the week (M-W-F or Tu-Th), some classes may not
2855 meet on Friday, or may have a tutorial section one day a week. The registrar's system, ever mindful of
2856 student privacy, shares only minimal information with the building systems such as how many students
2857 will be supported.

2858 The Registrar's system schedule building systems using the Calendar Gluon (registrar's information) and
2859 the student counts for each interval, and schedules the Sequence in classroom schedule 1 three days a
2860 week for the next 10 weeks. The Registrar's system also schedules the sequence in classroom schedule
2861 2 two days a week, also for 10 weeks.

2862 This example demonstrates a system (A) that offers services using either of two sequences. Another
2863 business system (B) with minimal knowledge of how (A) works determines the performance requirements
2864 for (A). The business system (B) communicates these expectations are by scheduling the Sequences
2865 offered by (A).

2866 C.1.2 Market Performance schedule

2867 A factory relies on an energy-intensive process which is performs twice a year for eight weeks. The
2868 factory has some flexibility about scheduling the process; it can perform the work in either the early
2869 morning or the early evening; it avoids the afternoon when energy costs are highest. The factory works up
2870 a detailed profile of when it will need energy to support this process.



2871
2872 *Figure 6: Daily Load Profile for Market Operations Example*
2873 Factory management has decided that they want to use only renewable energy products for this process.
2874 They approach two regional wind farms with the intent of making committed purchases of wind energy.
2875 The wind farms consider their proposals taking into account the seasonal weather forecasts they use to
2876 project their weather capacity, and considering the costs that may be required to buy additional wind
2877 energy on the spot market to make up any shortfalls.

2878 Each energy supplier submits of the same sequence, a schedule, i.e. a daily starting time, and a price for
2879 the season's production. After considering the bids, and other internal costs of each proposal, the factory
2880 opts to accept a contract for the purchase of a fixed load profile (Partition), using the evening wind
2881 generation from one of the suppliers. This contract specifies Schedules of load purchases (starting data
2882 and time for the sequence) for each day.

2883

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.

2886
2887
