

WS-Calendar Version 1.0

Committee Specification Draft 04

17 June 2011

Specification URIs:

This version:

<http://docs.oasis-open.org/ws-calendar/ws-calendar-spec/v1.0/csd04/ws-calendar-spec-v1.0-csd04.pdf> (Authoritative)
<http://docs.oasis-open.org/ws-calendar/ws-calendar-spec/v1.0/csd04/ws-calendar-spec-v1.0-csd04.html>
<http://docs.oasis-open.org/ws-calendar/ws-calendar-spec/v1.0/csd04/ws-calendar-spec-v1.0-csd04.doc>

Previous version:

<http://docs.oasis-open.org/ws-calendar/ws-calendar-spec/v1.0/csd03/ws-calendar-spec-v1.0-csd03.pdf> (Authoritative)
<http://docs.oasis-open.org/ws-calendar/ws-calendar-spec/v1.0/csd03/ws-calendar-spec-v1.0-csd03.html>
<http://docs.oasis-open.org/ws-calendar/ws-calendar-spec/v1.0/csd03/ws-calendar-spec-v1.0-csd03.doc>

Latest version:

<http://docs.oasis-open.org/ws-calendar/ws-calendar/v1.0/ws-calendar-1.0-spec.pdf> (Authoritative)
<http://docs.oasis-open.org/ws-calendar/ws-calendar/v1.0/ws-calendar-1.0-spec.html>
<http://docs.oasis-open.org/ws-calendar/ws-calendar/v1.0/ws-calendar-1.0-spec.doc>

Technical Committee:

OASIS Web Services Calendar (WS-Calendar) TC

Chair:

[Toby Considine](#)

Editors:

[Toby Considine](#)
[Mike Douglass](#)

Related work:

This specification is related to:

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

XML schemas: [ws-calendar-spec/v1.0/csd04/xsd/](#)

Abstract:

WS-Calendar describes

- A semantic (or information) model for exchange of calendar information to coordinate activities
- A means of synchronizing and maintaining calendars

The specification includes XML vocabularies for the interoperable and standard exchange of:

- Schedules, including sequences of schedules
- Intervals, including sequences of Intervals
- Other calendar information consistent with the IETF iCalendar standards

These vocabularies describe schedules and Intervals future, present, or past (historical).

The specification is divided into three parts.

- 1) The information model and XML vocabularies for exchanging schedule information
- 2) RESTful Services for calendar update and synchronization
- 3) Web services for calendar update and synchronization

The Technical Committee has decided not to publish Parts 2 and 3 until a later version.

Status:

This document was last revised or approved by the OASIS Web Services Calendar (WS-Calendar) TC on the above date. The level of approval is also listed above. Check the "Latest version" location noted above for possible later revisions of this document.

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

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

Citation format:

When referencing this specification the following citation format should be used:

[WS-Calendar]

WS-Calendar Version 1.0. 17 June 2011. OASIS Committee Specification Draft 04.
<http://docs.oasis-open.org/ws-calendar/ws-calendar-spec/v1.0/csd04/ws-calendar-spec-v1.0-csd04.pdf>.

Notices

Copyright © OASIS Open 2011. All Rights Reserved.

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

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

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

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

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

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

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

The names "OASIS" and "WS-Calendar" are trademarks of [OASIS](#), the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see <http://www.oasis-open.org/who/trademark.php> for above guidance.

Table of Contents

1	Introduction	7
1.1	Terminology	8
1.2	Normative References	8
1.3	Non-Normative References	9
1.4	Contributions	10
1.5	Namespace	10
1.6	Naming Conventions	11
1.7	Editing Conventions	11
1.8	Architectural References	11
1.9	Semantics	11
2	Overview of WS-Calendar	16
2.1	Approach taken by the WS-Calendar Technical Committee	16
2.2	Communicating Schedules and Service Performance	16
2.2.1	Which Time? UTC vs. Local Time	17
2.3	Overview of This Document	17
2.4	Security Considerations	18
3	PART ONE: Information model for WS-Calendar	19
3.1	Intervals, Temporal Relations, and Sequences	19
3.1.1	Core Semantics derived from [XCAL]	19
3.1.2	Intervals	20
3.1.3	Connecting the Intervals	21
3.1.4	Sequences: Combining Intervals	23
3.1.5	State Changes	25
3.2	Attachments and Tolerance	26
3.2.1	Attachment and the Artifact	26
3.2.2	Tolerance: What is Timely Performance	27
3.3	Using Sequences: referencing, modifying, and remote access	29
3.3.1	References and Inheritance	29
3.3.2	Gluons and Sequences	31
3.3.3	Inheritance rules for Gluons	33
3.3.4	Optimizing the expression of a Partition	34
3.3.5	Notifying Partners of Process Availability	36
3.3.6	Other Scheduling Scenarios	39
3.4	Time Stamps	41
3.4.1	Time Stamp Realm Discussion	42
4	Conformance and Rules for WS-Calendar and Referencing Specifications	43
4.1	Introduction	43
4.2	Conformance Rules for WS-Calendar	43
4.2.1	Inheritance in WS-Calendar	43
4.2.2	Specific Attribute Inheritance	43
4.2.3	General Conformance Issues	44
4.2.4	Covarying Elements	44
4.2.5	Conformance of Intervals	44

4.2.6 Conformance of Bound Intervals and Sequences.....	45
4.3 Conformance Rules for Specifications Claiming Conformance to WS-Calendar.....	45
4.4 Security Considerations.....	45
A. Acknowledgements	46
B. An Introduction to Internet Calendaring.....	47
B.1 icalendar	47
B.1.1 History	47
B.1.2 Data model.....	47
B.1.3 Scheduling	48
B.1.4 Extensibility	48
B.2 Calendar data access and exchange protocols	48
B.2.1 Internet Calendar Subscriptions.....	48
B.2.2 CalDAV	48
B.2.3 ActiveSync/SyncML	49
B.2.4 CalWS.....	49
B.2.5 iSchedule	49
B.3 References	49
C. Overview of WS-Calendar, its Antecedents and its Use	50
C.1 Scheduling Sequences	51
C.1.1 Academic Scheduling example.....	51
C.1.2 Market Performance schedule.....	52
D. Revision History.....	53

Tables

Index of Tables

Table 1-1: Namespaces used in this specification	10
Table 1-2: Schemas and Extensions Used in this Specification	10
Table 1-3: Semantics: Foundational Elements	12
Table 1-4: Semantics: Relations, Limits, and Constraints	12
Table 1-5: Semantics: Inheritance	13
Table 1-6: Semantics: Describing Intervals	14
Table 3-1: Elements of Intervals	20
Table 3-2: Temporal Relationships	21
Table 3-3: Elements of a WS-Calendar Attachment	26
Table 3-4: Tolerance Elements	27
Table 3-5: Gluon Elements	30
Table 3-6 Gluon Inheritance rules	33
Table 3-7: Elements of Time Stamps	41

Index of Examples

Example 3-1: An Unanchored Interval	21
Example 3-2: Temporal Relationship	22
Example 3-3: Temporal Relationship with Gap	22
Example 3-4: Temporal Relationship without Gap	22
Example 3-5: Introducing the Sequence	23
Example 3-6: An Anchored Sequence	24
Example 3-7 State Change communication using Zero Duration Interval	25
Example 3-8 State Change communication using Event without Duration or End	25
Example 3-9: Use of an Attachment with inline XML artifact	26
Example 3-10: Use of an Attachment with external reference	27
Example 3-11: Interval with inline XML artifact and optional specified Performance	28
Example 3-12: Sequence with Performance defined in the Gluon	31
Example 3-13: Partition with Duration and Relationship defined in the Gluon	34
Example 3-14: Vavailability	37
Example 3-15 Gluon publishing availability of pre-existing sequence	38
Example 3-16: Standard Sequence with Ramp-Up and Ramp Down	39

1 Introduction

2 The information model of WS-Calendar is intended to be used to define information payloads for Web
3 services and Service-style interactions **[SOA-RM]**. Placing these requirements in context requires a brief
4 overview of service requirements.

5 Agreement on when something should or did occur is fundamental to negotiating service use. Negotiated
6 services must be audited to understand timely performance. Short running services traditionally have
7 been handled as if they were instantaneous, and have handled scheduling through just-in-time requests.
8 Longer running processes, including physical processes, may require significant lead-times. When
9 multiple long-running services participate in the same business process, it may be more important to
10 negotiate a common completion time than a common start time. Pre-existing approaches that rely on
11 direct control of such services by a central system increases integration costs and reduce interoperability
12 as they require the controlling agent to know and manage multiple lead times.

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

19 Web services already coordinate a number of physical processes. Web services for building-based
20 systems include the standards **[oBIX]**, BACnet/WS¹ LON-WS², OPC UA³, as well as a number of
21 proprietary systems. The European research and advanced development project SIRENA (Service
22 Infrastructure for Real time Embedded Networked Applications) explored SOA for buildings, factories and
23 devices, including SODA (Service Oriented Device Architecture). SOA4D⁴ (Service-Oriented Architecture
24 for Devices) offers a collaborative open source development web platform, including implementations
25 **([SOAP] messaging, [WS-Management], [WS-Security], [DPWS])** adapted to the specific constraints of
26 embedded devices. There is a growing interest in coordinating the activities of things, building systems,
27 industrial processes, homes, with human enterprise activities. In particular, if building systems coordinate
28 with the schedules of the building's occupants, they can reduce energy use while improving performance.

29 An increasing number of specifications envision synchronization of processes through mechanisms
30 including broadcast scheduling. Efforts to build an intelligent power grid (or smart grid) rely on
31 coordinating processes in homes, offices, and industry with projected and actual power availability;
32 mechanisms proposed include communicating different prices at different times. These and other efforts
33 can benefit from a common cross-domain, cross specification standard for communicating schedule and
34 interval.

¹ BACnet® is a registered trademark of American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE).

² LON is a registered trademark of Echelon Corporation.

³ OPC UA is owned by the OPC Foundation

⁴ <http://cms.soa4d.org/>

35 For human interactions and human scheduling, the well-known iCalendar format addresses these
36 problems. Prior to WS-Calendar, there has been no comparable standard for web services. As an
37 increasing number of physical processes become managed by web services, the lack of a similar
38 standard for scheduling and coordination of services becomes critical.

39 The WS-Calendar Technical Committee (TC) based its work upon the iCalendar specification as updated
40 in 2009 (IETF [RFC5545] and its the XML serialization [XCAL], currently (2011-05) on a standards track
41 in the IETF. The specification adopts the semantics and vocabulary of iCalendar for application to the
42 completion of service contracts and inter-process interactions. Members of the Calendaring and
43 Scheduling Consortium (CalConnect.org) developed both updates to IETF specifications and provided
44 advice to this TC.

45 While this specification (WS-Calendar) defines the use of core semantic elements from iCalendar, no part
46 of this document is intended to prevent the use of other semantic elements from iCalendar from being
47 used. WS-Calendar describes the minimal use of that standard, not the maximal.

48 Everything with the exception of all examples, all appendices, and the introduction is normative unless
49 otherwise specifically noted.

50 1.1 Terminology

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

54 1.2 Normative References

- 55 **CalendarResource** C. Joy, C. Daboo, M Douglass, *Schema for representing resources for*
56 *calendaring and scheduling services*, [http://tools.ietf.org/html/draft-cal-resource-](http://tools.ietf.org/html/draft-cal-resource-schema-03)
57 [schema-03](http://tools.ietf.org/html/draft-cal-resource-schema-03), (Internet-Draft), November 2010.
- 58 **FreeBusy** E York. *Freebusy read URL*,
59 [http://www.calconnect.org/pubdocs/CD0903%20Freebusy%20Read%20URL%20](http://www.calconnect.org/pubdocs/CD0903%20Freebusy%20Read%20URL%20V1.0.pdf)
60 [V1.0.pdf](http://www.calconnect.org/pubdocs/CD0903%20Freebusy%20Read%20URL%20V1.0.pdf), April 2009
- 61 **REST** T Fielding, *Architectural Styles and the Design of Network-based Software*
62 *Architectures*, <http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>.,
63 *Dissertation*, 2000
- 64 **ISO8601** ISO (International Organization for Standardization). *Representations of dates*
65 *and times, third edition*, December 2004, (ISO 8601:2004)
- 66 **RFC2119** S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*,
67 <http://www.ietf.org/rfc/rfc2119.txt>, IETF RFC2119, March 1997.
- 68 **RFC2616** R Fielding, et al. et al, *Hypertext Transfer Protocol -- HTTP/1.1*
69 <http://tools.ietf.org/html/rfc2616>, IETF RFC2616, draft standard, June, 1999
- 70 **RFC3339** G Klyne, C Newman, *Date and Time on the Internet: Timestamps*
71 <http://tools.ietf.org/html/rfc3339>, IETF RFC3339 proposed standard, July 2002
- 72 **RFC4791** Daboo, et al. *Calendaring Extensions to WebDAV (CalDAV)*,
73 <http://www.ietf.org/rfc/rfc4791.txt>. IETF RFC 4791, proposed standard, March
74 2007
- 75 **RFC5545** B. Desruisseaux *Internet Calendaring and Scheduling Core Object Specification*
76 *(iCalendar)*, <http://www.ietf.org/rfc/rfc5545.txt>, IETF RFC5545, proposed
77 standard, September 2009
- 78 **RFC5546** C. Daboo *iCalendar Transport-Independent Interoperability Protocol (iTIP)*,
79 <http://www.ietf.org/rfc/rfc5546.txt>, IETF RFC5546, proposed standard, December
80 2009.
- 81 **RFC5988** M. Nottingham, *Web Linking*, <http://tools.ietf.org/html/rfc5988>, IETF RFC5988,
82 proposed standard, October 2010.

83	RFC6047	A. Melnikov, <i>iCalendar Message-Based Interoperability Protocol (iMIP)</i> , http://tools.ietf.org/html/rfc6047 , IETF RFC6047, December 2010.
84		
85	SOA-RM	OASIS Standard, <i>Reference Model for Service Oriented Architecture 1.0</i> , http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.pdf , October 2006.
86		
87	SOAP	M Gudgin, M Hadley, N Mendelsohn, J Moreau, H Nielsen, A Karmarkar, SOAP Version 1.2 Part 1: Messaging Framework, W3C Recommendation http://www.w3.org/TR/soap12-part1/ , W3C Recommendation, April 2007
88		
89		
90	Vavailability	C. Daboo, B. Desruisseaux, <i>Calendar Availability</i> , http://tools.ietf.org/html/draft-daboo-calendar-availability-02 , IETF Internet Draft, April 2011
91		
92	xCal	C. Daboo, M Douglass, S Lees <i>xCal: The XML format for iCalendar</i> , http://tools.ietf.org/html/draft-daboo-et-al-icalendar-in-xml-08 , IETF Internet-Draft, April 2011.
93		
94		
95	XPATH	A Berglund, S Boag, D Chamberlin, M Fernández, M Kay, J Robie, J Siméon, <i>XML Path Language (XPath) 2.0 (Second Edition)</i> , http://www.w3.org/TR/xpath20/ , W3C Recommendation, December 2010.
96		
97		
98	XLINK	S DeRose, E Maler, D Orchard, N Walsh, <i>XML Linking Language (XLink) Version 1.1</i> , http://www.w3.org/TR/xlink11/ May 2010.
99		
100	XPTR	P Grosso, E Maler, J Marsh, N Walsh, <i>XPointer Framework</i> , http://www.w3.org/TR/xptr-framework/ W3C Recommendation, March 2003
101		
102		P Grosso, E Maler, J Marsh, N Walsh, <i>XPointer element() Scheme</i> , http://www.w3.org/TR/xptr-element/ , W3C Recommendation March 2003
103		
104	XML NAMES	T Bray, D Hollander, A Layman, R Tobin, HS Thompson “Namespaces in XML 1.0 (Third Edition)” http://www.w3.org/TR/xml-names/ W3C Recommendation, December 2009
105		
106		
107	XML SCHEMA	PV Biron, A Malhotra, <i>XML Schema Part 2: Datatypes Second Edition</i> , http://www.w3.org/TR/xmlschema-2/ October 2004.
108		
109	XRD	OASIS XRI Committee Draft 01, <i>Extensible Resource Descriptor (XRD) Version 1.0</i> , http://docs.oasis-open.org/xri/xrd/v1.0/cd01/xrd-1.0-cd01.pdf October 2009.
110		

111 1.3 Non-Normative References

112	DPWS	T Nixon, A Regnier, D Driscoll, Antoine Mensch , <i>Devices Profile for Web Services 1.1</i> , http://docs.oasis-open.org/ws-dd/ns/dpws/2009/01 , July 2009
113		
114	NIST Framework	NIST Framework and Roadmap for Smart Grid Interoperability Standards , Office of the National Coordinator for Smart Grid Interoperability, Release 1.0, NIST Special Publication 1108, http://www.nist.gov/public_affairs/releases/upload/smartgrid_interoperability_final.pdf .
115		
116		
117		
118		
119	NAESB Requirements	NAESB Smart Grid Requirements (awaiting publication) (draft contributed) http://lists.oasis-open.org/archives/ws-calendar-comment/201005/doc00000.doc , May 2010
120		
121		
122	RFC4918	L. Dusseault, <i>HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV)</i> . http://tools.ietf.org/html/rfc4918 IETF RFC 4918, proposed standard, June 2007
123		
124		
125	TZDB	P Eggert, A.D. Olson, "Sources for Time Zone and Daylight Saving Time Data", http://www.twinsun.com/tz/tz-link.htm
126		
127	Time Zone Recommendations , CalConnect, <i>CalConnect EDST (Extended Daylight Savings Time) Reflections and Recommendations</i> , Version: 1.1, http://www.calconnect.org/pubdocs/CD0707%20CalConnect%20EDST%20Reflections%20and%20Recommendations%20V1.1.pdf	
128		
129		
130		
131		October 2010
132	Time Zone Service , M Douglas, C Daboo, <i>Timezone Service Protocol</i> , Draft RFC,IETF, http://datatracker.ietf.org/doc/draft-douglass-timezone-service/	
133		

- 134 **WS-Management** DMTF Standard, **Web Services for Management**,
 135 http://www.dmtf.org/sites/default/files/standards/documents/DSP0226_1.0.0.pdf
 136 **April 2010**
- 137 **WS-Security** K Lawrence, C Kaler, A Nadalin, R Monzillo, P Hallam-Baker, **Web Services**
 138 **Security: 4 SOAP Message Security 1.1**, [http://www.oasis-](http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf)
 139 [open.org/committees/download.php/16790/wss-v1.1-spec-os-](http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf)
 140 [SOAPMessageSecurity.pdf](http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf), **February 2006**
- 141 **WSDL** R Chinnici, J Moreau, A Ryman, S Weerawarana, *Web Services Description*
 142 *Language (WSDL) Version 2.0 Part 1: Core Language*
 143 <http://www.w3.org/TR/wsdl20/>, W3C Recommendation, June 2007.
 144 R Chinnici, H Haas, A Lewis, J Moreau, D Orchard, S Weerawarana, *Web*
 145 *Services Description Language (WSDL) Version 2.0 Part 2: Adjuncts*,
 146 <http://www.w3.org/TR/wsdl20-adjuncts/>, W3C Recommendation, June 2007.

147 **1.4 Contributions**

148 The NIST Roadmap for Smart Grid Interoperability Standards [**NIST Framework**] requested that many
 149 standards development organizations (SDOs) and trade associations work together closely in
 150 unprecedented ways. An extraordinary number of groups came together and contributed effort, and time,
 151 requirements, and documents. The North American Energy Standards Board (NAESB) oversaw meetings
 152 with many representatives from every energy sector to contribute requirements to the TC. These
 153 meetings were presided over by Jonathan Booe to support the Roadmap’s Priority Action Plan 04
 154 (PAP04), a common specification of time and schedule.

155 **NAESB Smart Grid Standards Development Subcommittee:**

156 The following documents are password protected. For information about obtaining access to these
 157 documents, please visit www.naesb.org or contact the NAESB office at (713) 356 0060.

158 **Wholesale** http://www.naesb.org/member_login_check.asp?doc=fa_2010_weq_api_6_b_ii.doc

159 **Retail** http://www.naesb.org/member_login_check.asp?doc=fa_2010_retail_api_9_b_ii.doc

160 **1.5 Namespace**

161 The XML namespace [**XMLNAMES**] URI that MUST be used by implementations of this specification is:

```
162 urn:ietf:params:xml:ns:icalendar-2.0
```

163 Table 1-1 lists the XML schemas that are used in this specification. The choice of any namespace prefix
 164 is arbitrary and not semantically significant.

165 *Table 1-1: Namespaces used in this specification*

Prefix	Namespace
xs	http://www.w3.org/2001/XMLSchema
xcal	urn:ietf:params:xml:ns:icalendar-2.0
ts	http://docs.oasis-open.org/ns/ws-calendar/timestamp/201103

166 The Resource Directory Description Language [**RDDL 2.0**] document that describes this namespace can
 167 be found at <http://docs.oasis-open.org/ns/ws-calendar>. The normative schemas for WS-Calendar can be
 168 found linked from this namespace document. The schemas are listed in Table 1-2.

169 *Table 1-2: Schemas and Extensions Used in this Specification*

Schema	Description
iCalendar.xsd	Base Schema expressing core iCalendar information
iCalendar-params.xsd	Parameters used in iCalendar objects

iCalendar-props.xsd	Properties of iCalendar objects
iCalendar-valtypes.xsd	Values used by iCalendar
iCalendar-link-extension.xsd	Link extensions based on web linking [RFC5998] to define relationships between Components.
iCalendar-wscal-extensions.xsd	Extensions to iCalendar to support service functionality
iCalendar-bw-extensions.xsd	Extensions to support integration with Bedeworks server.
iCalendar-ms-extensions.xsd	Extensions to support integration with MS Exchange Server
TimeStamp.xsd	An ancillary information model describing the elements needed to support event forensics

170 Reviewers can find the schemas at <http://docs.oasis-open.org/ws-calendar/ws-calendar-spec/v1.0/csd03/xsd/>.
171

172 1.6 Naming Conventions

173 This specification follows some naming conventions for artifacts defined by the specification, as follows:

174 For the names of elements and the names of attributes within XSD files, the names follow the lower
175 camelCase convention, with all names starting with a lower case letter. For example,

```
176 <element name="componentType" type="energyinterop:ComponentType"/>
```

177 For the names of types within XSD files, the names follow the lower CamelCase convention with all
178 names starting with a lower case letter prefixed by "type-". For example,

```
179 <complexType name="type-componentService">
```

180 For the names of intents, the names follow the lower camelCase convention, with all names starting with
181 a lower case letter, EXCEPT for cases where the intent represents an established acronym, in which
182 case the entire name is in upper case.

183 An example of an intent that is an acronym is the "SOAP" intent.

184 1.7 Editing Conventions

185 For readability, element names in tables appear as separate words. The actual names are
186 lowerCamelCase, as specified above, and as they appear in the XML schemas.

187 All elements in the tables not marked as "optional" are mandatory.

188 Information in the "Specification" column of the tables is normative. Information appearing in the note
189 column is explanatory and non-normative.

190 All sections explicitly noted as examples are informational and are not to be considered normative.

191 1.8 Architectural References

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

195 1.9 Semantics

196 Certain terms appear throughout this document, some with extensive definitions. Table 1-3 provides
197 definitions for the convenience of the reader and reviewer. Many terms require fuller discussion than is in
198 this section, and are discussed in greater depth in later sections. In all cases, the normative actual
199 definition is the one in this section.

200 WS-Calendar terminology begins with a specialized terminology for the segments of time, and for groups
 201 of related segments of time. These terms are defined in Table 1-3 through Table 1-6 below.

202 *Table 1-3: Semantics: Foundational Elements*

Time Segment	Definition
Component	In iCalendar, the primary information structure is a Component. Intervals and Gluons are new Components defined in this specification.
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.
Interval	The Interval is a single Duration derived from the common calendar Components as defined in iCalendar ([RFC5545]). An Interval is part of a Sequence. An entire Sequence can be scheduled by scheduling a single Interval in that sequence. For this reason, Intervals are defined through Duration rather than through dtStart or dtEnd.
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. A Sequence may optionally include a Lineage. A Sequence can be scheduled multiple times through repeated reference by different Gluons. Intervals are defined through their Duration, and the schedule, dtEnd or dtStart, is applied to the Sequence as a whole.
Partition	A Partition is a set of consecutive Intervals. The Partition includes the trivial case of a single Interval. Partitions are used to define a single service or behavior that varies over time. Examples include energy prices over time and energy usage over time.
Gluon	A gluon influences the serialization of Intervals in a Sequence, though inheritance and through schedule setting. The Gluon is similar to the Interval, but has no service or schedule effects until applied to an Interval or Sequence.
Artifact	An Artifact is the thing that occurs during an Interval. WS-Calendar uses the Artifact as a placeholder. The contents of the Artifact are not specified in WS-Calendar; rather the Artifact provides an extension base for the use of WS-Calendar in other specifications. Artifacts may inherit elements as do Intervals within a Sequence.

203 WS-Calendar works with groups of Intervals that have relationships between them. These relations
 204 constrain the final instantiation of a schedule-based service. Relations can control the ordering of
 205 Intervals in a Sequence. They can describe when a service can be, or is prevented from, being invoked.
 206 They establish the parameters for how information will be shared between elements using Inheritance.
 207 The terminology for these relationships is defined in Table 1-4.

208 *Table 1-4: Semantics: Relations, Limits, and Constraints*

Term	Definition
Link	The Link is used by one WS-Calendar object to reference another. A link can reference either an internal object, within the same calendar, or an external object in a remote system.

Term	Definition
Relationship	Relationships link between Components for Binding. ICalendar defines several relationships, but WS-Calendar uses only the CHILD relationship, and that only to bind Gluons to each other and to Intervals.
Temporal Relationship	Temporal Relationships extend the [RFC5545] Relationships to define how Intervals become a Sequence by creating an order between Intervals. The Predecessor Interval includes a Temporal Relation, which references the Successor Interval. When the start time and Duration of one Interval is known, the start time of the others can be computed through applying Temporal Relations.
Availability	Availability expresses the range of times in which an Interval or Sequence can be Scheduled. Availability often overlays or is overlaid by Busy. Availability can be Inherited.
Busy	Busy expresses the range of times in which an Interval or Sequence cannot be Scheduled. Busy often overlays is overlaid by Availability. Busy can be Inherited.
Child, Children	The CHILD relationship type (rel_type) defines a logical link (via URI or UID) from parent object to a child object. A Child object is the target of one or more CHILD relationships and may have one to many Parent objects.
Parent [Gluon]	A Gluon (in a Sequence) that includes a CHILD relationship parameter type (rel_type) defines a logical link (via URI or UID) from parent object to a child object. A Parent Component contains one or more CHILD Relationships

209 WS-Calendar describes how to modify and complete the specification of Sequences. WS-Calendar calls
210 this process Inheritance and specifies a number of rules that govern inheritance. Table 1-5 defines the
211 terms used to describe inheritance.

212 *Table 1-5: Semantics: Inheritance*

Term	Definition
Lineage	The ordered set of Parents that results in a given inheritance or execution context for a Sequence.
Inheritance	Parents bequeath information to Children that inherit them. If a child does not already possess that information, then it accepts the inheritance. WS-Calendar specifies rules whereby information specified in one informational object is considered present in another that is itself lacking expression of that information. This information is termed the Inheritance of that object.
Bequeath	A Parent Bequeaths attributes (Inheritance) to its Children.
Inherit	A Child Inherits attributes (Inheritance) from its Parent.
Covarying Attributes	Some attributes are inherited as a group. If any member of that group is expressed in a Child, all members of that group are deemed expressed in that Child, albeit some may be default values. These characteristics are called covarying or covariant. A parent bequeaths covarying characteristics as a group and a child accepts or refuses them as a group.
Decouplable Attributes	Antonym for Covarying Attributes. Decouplable Attributes can be inherited separately.

213 As Intervals are processed, as Intervals are assembled, and as inheritance is processed, the information
 214 conveyed about each element changes. When WS-Calendar is used to describe a business process or
 215 service, it may pass through several stages in which the information is not yet complete or actionable, but
 216 is still a conforming expression of time and Sequence. Table 1-6 defines the terms used when discussing
 217 the processing or processability of Intervals and Sequences.

218 During the life-cycle of communications concerning Intervals, different information may be available or
 219 required. For service performance, Start Duration and the Attachment Payload must be complete. These
 220 may not be available or required during service advertisement or other pre-execution processes. Table
 221 1-6 defines the language used to discuss how the information in an Interval is completed.

222 *Table 1-6: Semantics: Describing Intervals*

Term	Definition
Designated Interval	An Interval that is referenced by a Gluon is the Designated Interval for a Series. An Interval can be Designated and still not Anchored.
Anchored	An Interval is Anchored when it includes a Start or End, either directly or through Binding. A Sequence is Anchored when its Designated Interval is Anchored.
Unanchored	An Interval is Unanchored when it includes neither a Start or an End, either internally, or through Binding. A Sequence is Unanchored if its Designated Interval Unanchored. <i>Note: a Sequence that is re-used may be Unanchored in one context even while it is Anchored in another.</i>
Binding	Binding is the application of information to an Interval or Gluon, information derived through Inheritance or through Temporal Assignment.
Bound Element	A Bound Element refers to an Element and its Value after Binding, e.g., a Bound Duration.
Bound Interval	A Bound Interval refers to an Interval and the values of its Elements after Binding.
Bound Sequence	A Bound Sequence refers to a Sequence and the values of its Intervals after Binding.
Partially Bound	Partially Bound refers to an Interval or a Sequence which is not yet complete following Binding, i.e., the processes cannot yet be executed.
Fully Bound	Fully Bound refers to an Interval or Sequence that is complete after Binding, i.e., the process can be unambiguously executed when Anchored.
Unbound	An Unbound Interval or Sequence is not itself complete, but must still receive inheritance to be fully specified. A Sequence or Partition is Unbound if it contains at least one Interval that is Unbound.
Constrained	An Interval is Constrained if it is not Anchored and it is bound to one or more Availability or Free/Busy elements
Temporal Assignment	Temporal Assignment determines the start times of Intervals in a Sequence through processing of their Durations and Temporal Relations.
Scheduled	A Sequence or Partition is said to be Scheduled when it is Anchored, Fully Bound, and service performance has been requested.

Term	Definition
Unscheduled	An Interval is Unscheduled if it is not Anchored, nor is any Interval in its Sequence Anchored. A Sequence or Partition is Unscheduled if none of its Intervals, when Fully Bound, is Scheduled.
Predecessor Interval	A Predecessor Interval includes a Temporal Relation which references a Successor Interval.
Successor Interval	A Successor Interval is one referred to by a Temporal Relationship in a Predecessor Interval.
Antecedent Interval(s)	Antecedents are an Interval or set of Intervals that precede a given Interval within the same Sequence
Earliest Interval	The set of Intervals at the earliest time in a given Sequence
Composed Interval	A Composed Interval is the virtual Interval specified by applying inheritance through the entire lineage and into the Sequence in accord with the inheritance rules. A Composed Interval may be Bound, Partially Bound, or Unbound.
Composed Sequence	A Composed Sequence is the virtual Sequence specified by applying inheritance through the entire lineage and into the Sequence in accord with the inheritance rules. A Composed Sequence may be Bound, Partially Bound, or Unbound.
Comparable Sequences	Two Sequences are Comparable if and only if the Composed version of each defines the same schedule.

2 Overview of WS-Calendar

223

224 A calendar communication without a real world effect is of little interest. That real world effect is the result
225 of a service execution context within a policy context. Practitioners can use WS-Calendar to add
226 communication of schedule and Interval to the execution context of a service. Use of WS-Calendar will
227 align the performance expectations between execution contexts in different domains. The Technical
228 Committee intends for other specifications and standards to normatively reference and claim
229 conformance to WS-Calendar, bringing a common scheduling context to diverse interactions in different
230 domains

2.1 Approach taken by the WS-Calendar Technical Committee

231

232 The Technical Committee (TC) based its work upon the iCalendar specification as updated in 2009 (IETF
233 **[RFC5545]** and its the XML serialization **[XCAL]**, currently (2011-05) on a standards track in the IETF.
234 Members of the Calendaring and Scheduling Consortium (CalConnect.org) developed both updates to
235 IETF specifications and provided advice to this TC. **[RFC5545]** provides the normative vocabulary for use
236 in this specification.

237 This committee developed the normative schema (XSD) for iCalendar. This schema, including the
238 schema extensions necessary for the services defined herein, is part of the WS-Calendar specification.

239 The committee solicited requirements from a range of interests, notably the NIST Smart Grid Roadmap
240 **[NIST Framework]** and the requirements of the Smart Grid Interoperability Panel (SGIP) as developed by
241 the North American Energy Standards Board (NAESB) **[NAESB Requirements]**. Others submitting
242 requirements included members of the oBIX technical committee and representatives of the FIX Protocol
243 Association. These requirements are reflected in the semantic elements described in Chapters 3 and 4.

244 In a parallel effort, the CalConnect TC-XML committee developed a number of schedule and calendar-
245 related services. CalConnect drew on its experience in interoperability between enterprise calendaring
246 systems as well as interactions with web-based calendars and personal digital assistants (PDAs). These
247 services were developed as RESTfull (using **[REST]**) services by CalConnect and contributed to the WS-
248 Calendar TC. CalConnect also developed and contributed **[SOAP]** and **[WSDL]** definitions to this TC.

2.2 Communicating Schedules and Service Performance

249

250 Time semantics are critical to process interactions. Services requested differently can have different
251 effects on performance even though they appear to request the same time interval. This is inherent in the
252 concept of a service-oriented architecture.

253 As defined in the OASIS Reference Model for Service Oriented Architecture 1.0 **[SOA-RM]**, service
254 requests access the capability of a remote system.

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

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

263 A request for remote service performance is a request for specific real world effects. For process
264 interaction, these effects are expected to occur during a given period. Consider two service providers that
265 offer the same service. One must start planning an hour or more in advance. The second may be able to
266 achieve the service in five minutes. The service start time is the time when that service becomes fully
267 available; that is the time specified in service interactions. Because this service start time and service

268 period are all that matters, the same service can be offered by different providers using quite different
269 technologies.

270 2.2.1 Which Time? UTC vs. Local Time

271 Coordinated Universal Time (abbreviated UTC) is a time standard based on International Atomic Time
272 (TAI) with leap seconds added at irregular intervals to compensate for the Earth's slowing rotation. Time
273 zones around the world can be expressed as positive or negative offsets from UTC.

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

280 As long as the rules never change this causes no confusion—but they do. Recent experience has
281 included considerable efforts when the rules for the start of Daylight Saving Time (DST) have changed. If
282 all information is in UTC, and no record of the event's basis in the local time and time zone remains, there
283 is no way to re-compute existing contracts. It is often necessary to know if UTC was calculated based on
284 an old or a new rule.

285 A triplet of Local time + timezoneid + (UTC or offset) always allows the determination if a time is valid. If a
286 recalculation of UTC for that local time + tzid results in a different value from that stored then presumably
287 the DST rules have changed since the data was stored. If one can detect that the scheduled time is no
288 longer valid, one can take corrective action.

289 The Technical Committee makes no representation as whether UTC or local time are more appropriate
290 for a given interaction. Because WS-Calendar is based on **[iCalendar]**, business practices built upon WS-
291 Calendar can support either. Specifications that claim conformance with this specification may require
292 choices to support their particular business processes.

293 For a fuller discussion of time zones, consult **[Time Service Recommendations]** and **[Time Zone**
294 **Service]** in the non-normative references.

295 2.3 Overview of This Document

296 The specification consists of a standard schema and semantics for schedule and interval information.
297 Often the most important service schedule communications involve series of related services over time,
298 which WS-Calendar defines as a Sequence. These semantic elements are defined and discussed in
299 Section 3. While this specification describes only the use of core semantic elements from iCalendar, no
300 part of this document prevents other semantic elements from iCalendar from also being used.

301 Section 3.2 introduces notions of tolerance, i.e. what does it mean to be “on time”. This section also
302 describes the different ways to associate a service request with each Interval in a Sequence.

303 Managing information exchanges about a Sequence of events can easily become cumbersome, or prone
304 to error. WS-Calendar defines the Gluon, a mechanism for making assertions about all or most of the
305 Intervals in a Sequence. Intervals can inherit from a Gluon, or they can override locally assertions
306 inherited from the Gluon. Section 3.3 discusses inheritance and parsimony of communication and
307 introduces contract scheduling.

308 The practitioner must decide whether to use one or the other of these communication protocols, or
309 whether WS-Calendar artifacts are better used when embedded within other messages. These decisions
310 must be based upon the specific application and message content. Specifications that claim conformance
311 to this specification may wish to provide guidance appropriate for the business purposes of that
312 specification.

313 **2.4 Security Considerations**

314 **Part 1** describes an information model. The information models can be expressed in any interaction,
315 using any protocol. There are no security aspects of the information model.

316 Specifications which claim conformance with WS-Calendar may wish to specify security approaches or
317 techniques. Security choices must be based on the business requirements and operational risks of the
318 interaction that those specifications define. As this specification defines a general information model, for
319 use in many interactions, it specifies no security approach.

320 3 PART ONE: Information model for WS-Calendar

321 3.1 Intervals, Temporal Relations, and Sequences

322 WS-Calendar Elements are semantic elements derived from the **[XCAL]** specification. This set of
323 elements is smaller than those needed for full schedule interaction, and describe the Intervals, Durations,
324 and time-related events that are relevant to service interactions. WS-Calendar uses the elements to build
325 a precise vocabulary of time, Duration, Sequence, and Schedule.

326 WS-Calendar elements adapt the iCalendar objects to make interaction requirements explicit. For
327 example, in human schedule interactions, different organizations have their own expectations. Meetings
328 may start on the hour or within 5 minutes of the hour. As agents scheduled in those organizations, people
329 learn the expected precision. This precision expectation must be explicit to prevent interoperation
330 problems. This specification defines a performance element to elaborate the simple specification of
331 **[XCAL]** to make explicit the performance expectations within a scheduled event.

332 This specification defines common semantics for recording and exchanging event information (Time
333 Stamps).

334 3.1.1 Core Semantics derived from **[XCAL]**

335 The iCalendar data format **[RFC5545]** is a widely deployed interchange format for calendaring and
336 schedule data. The **[XCAL]** specification standardizes the XML representation of iCalendar information.
337 WS-Calendar relies on **[XCAL]** standards and data representation to develop its semantic Components.

338 3.1.1.1 Time

339 **[ISO8601]** defines string formats for the expression of date, time, and duration. **[ISO8601]** also defines
340 string formats to express the passage of time, herein a Duration. This specification relies extensively on
341 **[ISO8601]**. Examples of date and time representations include:

```
342 Year:  
343   YYYY (eg 1997)  
344 Year and month:  
345   YYYY-MM (eg 1997-07)  
346 Complete date:  
347   YYYY-MM-DD (eg 1997-07-16)  
348 Complete date plus hours and minutes:  
349   YYYY-MM-DDThh:mmTZD (eg 1997-07-16T19:20+01:00)  
350 Complete date plus hours, minutes and seconds:  
351   YYYY-MM-DDThh:mm:ssTZD (eg 1997-07-16T19:20:30+01:00)  
352 Complete date plus hours, minutes, seconds and a decimal fraction of a second  
353   YYYY-MM-DDThh:mm:ss.sTZD (eg 1997-07-16T19:20:30.45+01:00)
```

354 This specification is general purpose. Standards that claim conformance to this specification may need to
355 restrict the variability above to improve interoperation within their own interactions.

356 3.1.1.2 The iCalendar Components (VComponents)

357 iCalendar and **[XCAL]** have a number of long defined Component objects that comprise the payload
358 inside of an iCalendar message. These include the VTODO, the VALARM, the VEVENT. (The “v” that
359 begins each element name is there for historic reasons.) The definitions and use of each of the
360 vComponents can be found in **[RFC5545]**.

361 The vComponents share the same parameters and properties. The distinctions between these
362 informational types are ones of purpose and conformance. The Interval and Gluon are new
363 vComponents; each is derived from the same base type as the other vComponents.

364 This specification in no way deprecates the pre-existing vComponents. The new components are
 365 introduced to support stored sequences of operations and remote invocation. The existing vComponents
 366 are extended to support informational payloads for process interaction. A conforming specification can
 367 use both old and new vComponents where each makes sense.

368 The RESTful and SOAP in Parts Two and Three of a future version of this specification support all
 369 traditional vComponents as well as the new ones defined here. Conforming information elements MAY be
 370 processed using traditional iCalendar-based interactions (CalDAV, et al.) and managed in traditional
 371 iCalendar stores.

372 3.1.1.3 Duration and the Granularity of Time

373 This specification uses Duration as defined in **[ISO 8601]** as a data-type throughout. iCalendar makes a
 374 number of assumptions about the meaning of time when expressed as duration, i.e., a duration is over
 375 when the same common metric is reached in the next such unit. For example, a duration of one day
 376 starting at 6:00 AM lasts until 6:00 AM the next day. This becomes important during periods when the
 377 meaning of a duration changes. The passage of a month that begins on January 5 is complete on
 378 February 5. Another month comes to March 5. Each is expressed using the format “P1M”. These
 379 durations are, respectively, 31, 28 or 29, and 31 days. In a similar way, Years “P1Y” may be 365 or 366
 380 days long, days “P1D” may be 23, 24, or 25 hours long.

381 If the intention is to express 30 days, then one should use “P30D” and not “P1M”. Similarly, if the intent is
 382 to express from now until the same time tomorrow, use “P1D” rather than 24 hours “PT24H”.

383 3.1.2 Intervals

384 Clear communication of the continuous passage of time is critical to defining service coordination.

385 The building block for this information model is the Interval. The Interval is a time segment whose length
 386 is specified by a Duration. The Interval is a unit of time, and can be bound to service delivery. An
 387 Unscheduled Interval has no specified date and time. A Scheduled Interval has a specified start date and
 388 time. Intervals can legally contain all elements properties as defined in **[RFC5545]**. For convenience, the
 389 elements essential to coordinating service operations using Intervals are listed in Table 3-1.

390 An Interval is part of a Sequence. An entire Sequence can be scheduled by scheduling a single Interval in
 391 a Sequence. A single Sequence can be scheduled multiple times through repeated reference by different
 392 Gluons. It may be useful to consider the Unanchored Sequence as a process subroutine and that a Gluon
 393 can be used to invoke that subroutine. For this reason, of the three primary temporal elements (dtStart,
 394 dtEnd, and Duration) in a Component, the Duration has primacy in Intervals. Within a Sequence, a
 395 maximum of a single Interval MAY have a dtStart or a dtEnd.

396 Nothing in this section supersedes **[RFC5545]**. Implementers SHALL refer to those respective
 397 specifications **[RFC5545]** and the **[XCAL]** specifications for the normative description of each element
 398 with the exception of Duration, which is as defined as in **[ISO8601]**.

399 *Table 3-1: Elements of Intervals*

Elements	Description
Dtstamp	Identifies when Interval object was created
Uid	Used to enable unambiguous referencing by other Components
Duration	Identifies length of time for Interval. Duration must be known before an Interval can be transacted, but the Duration may only come through Binding.
DtStart	Scheduled start date and time for Interval. The Start must be known before an Interval can be transacted, but the Duration may only come through Binding.
Attach	In [XCAL] , any attachment. In WS-Calendar, the Attach contains the informational payload used by conforming specifications. See section 3.2.

Elements	Description
Relation	Relations contain the temporal relations between Intervals that create Sequences. Section 3.1.3. describes Temporal Relations and their use.

400 An Interval specifies how long an activity lasts. The example below (Example 3-1) shows a fragment of a
401 WS-Calendar-based message containing a single Unanchored Interval, i.e., it contains neither a dtStart
402 nor a dtEnd. Note that there is no Relationship; there is no need for Relationships until an Interval is
403 incorporated into a Sequence.

404 *Example 3-1: An Unanchored Interval*

```

405 <xcal:interval>
406   <xcal:properties>
407     <xcal:uid>
408       <xcal:text>bfe8069a-0c94-4b53-852f-6a2f12639c7e</xcal:text>
409     </xcal:uid>
410     <xcal:duration>
411       <xcal:duration>PT10H</xcal:duration>
412     </xcal:duration>
413   </xcal:properties>
414 </xcal:interval>

```

415 3.1.3 Connecting the Intervals

416 Many iCalendar communications involve more than one Interval. Classic iCalendar [RFC5545] defines
417 relationships internally. [xICAL] uses the extensible expression pattern of Web Links (as described in
418 [RFC5588]) to express the iCalendar relationships PARENT, CHILD, and SIBLING. This specification
419 extends these relationships by adding Temporal Relations. Temporal Relations consist of a reference, a
420 relation, and a Gap that specifies any Duration between Predecessor and Successor.

421 Unlike most semantic elements in this specification, Temporal Relations are defined in this specification,
422 rather than defined elsewhere and used herein.

423 *Table 3-2: Temporal Relationships*

Temporal Relationship	Short Form	Definition	Example
Gap		Duration indicating the time between the predecessor and the successor. Optional, where missing, Gap is treated as a zero duration	Gap may be positive or negative. In the examples below, the Gap, when present, is 20 minutes.
Finish To Start	FS	As soon as the predecessor Interval finishes, the successor Interval starts.	When sanding is complete, painting begins.
Finish To Finish	FF	The successor Interval continues as long as the predecessor Interval.	The concession stand stops serving 20 minutes after the end of the game.
Start To Finish	SF	The start of the predecessor controls the finish of the successor.	The start of Attendee Check-in controls the end of the Interval "Set up registration booth."
Start To Start	SS	The Predecessor Interval triggers the start of the second task.	20 minutes after the caterer begins work, the dining lines are open.

424 While simple relationships may be ordered based on which task occurs first (finishToStart), if a later
425 Interval is controlling, other choices may make more useful. For example, if ramp-up time must be
426 completed before run-time, and run-time start is indicated in a contract, it may be useful to specify that the

427 Ramp Interval (Successor) must complete before (startToFinish) the Designated Interval's (Predecessor)
428 scheduled start time. Specifications claiming conformance should consider statements of conformance
429 around Temporal Relationships.

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

432 *Example 3-2: Temporal Relationship*

```
433 <xcal:related-to>  
434   <xcal:parameters>  
435     <xcal:reltype>  
436       <xcal:text>FS</xcal:text>  
437     </xcal:reltype>  
438     <xcal:gap>  
439       <xcal:duration>PT10M</xcal:duration>  
440     </xcal:gap>  
441   </xcal:parameters>  
442   <xcal:uid>07fb177d-54ea-44ea-8ef5-5b763dc9f0c6</xcal:uid>  
443 </xcal:related-to>
```

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

447 *Example 3-3: Temporal Relationship with Gap*

```
448 <xcal:related-to>  
449   <xcal:parameters>  
450     <xcal:reltype>  
451       <xcal:text>FS</xcal:text>  
452     </xcal:reltype>  
453     <xcal:gap>  
454       <xcal:duration>PT10M</xcal:duration>  
455     </xcal:gap>  
456   </xcal:parameters>  
457   <xcal:uid>07fb177d-54ea-44ea-8ef5-5b763dc9f0c6</xcal:uid>  
458 </xcal:related-to>
```

459 Leaving out the optional Gap element, we have:

460 *Example 3-4: Temporal Relationship without Gap*

```
461 <xcal:related-to>  
462   <xcal:parameters>  
463     <xcal:reltype>  
464       <xcal:text>FS</xcal:text>  
465     </xcal:reltype>  
466   </xcal:parameters>  
467   <xcal:uid>07fb177d-54ea-44ea-8ef5-5b763dc9f0c6</xcal:uid>  
468 </xcal:related-to>
```

469 The expressions of a Temporal Relationship in Example 3-3 and Example 3-4 are equivalent.

470 Intervals with Temporal Relationships enable the message to express complex temporal relations to form
471 a Sequence. A Sequence consisting of identical consecutive Intervals is named a Partition. As the rules
472 for parsing XML do not mandate preservation of order within a sub-set, we cannot assume that order is
473 preserved when parsing a set of Intervals. For Sequences in WS-Calendar, then, mere order is not
474 enough—a Sequence is a collection of Intervals each of which Interval either refers to or is referred by at
475 least one Interval. Using the references, expressed as Temporal Relations, WS-Calendar describes a
476 single coherent Sequence assembled from a set of Intervals in a collection.

477 3.1.4 Sequences: Combining Intervals

478 A Sequence is a collection of Intervals with a coherent set of Temporal Relationships (Table 1-3).
479 Temporal Relationships are transitive, so that if Interval A is related to Interval B, and Interval B is related
480 to Interval C, then Interval A is related to Interval C. Sequences can also include Gluons (see section
481 3.3.1, *References and Inheritance.*, but for this section, we will discuss Sequences only as a set of
482 Intervals.

483 *Example 3-5: Introducing the Sequence*

```
484 <xcal:vcalendar>  
485   <xcal:components>  
486     <xcal:interval>  
487       <xcal:properties>  
488         <xcal:uid>  
489           <xcal:text>69343fc9-c1da-4cd0-abbd-  
490 889716a401d2</xcal:text>  
491         </xcal:uid>  
492         <xcal:duration>  
493           <xcal:duration>PT1H</xcal:duration>  
494         </xcal:duration>  
495       </xcal:properties>  
496     </xcal:interval>  
497     <xcal:interval>  
498       <xcal:properties>  
499         <xcal:uid>  
500           <xcal:text>0ba5a8c0-4eb2-49db-8514-  
501 5da18f53caaa</xcal:text>  
502         </xcal:uid>  
503         <xcal:related-to>  
504           <xcal:parameters>  
505             <xcal:reltype>  
506               <xcal:text>FS</xcal:text>  
507             </xcal:reltype>  
508           </xcal:parameters>  
509           <xcal:uid>69343fc9-c1da-4cd0-abbd-  
510 889716a401d2</xcal:uid>  
511         </xcal:related-to>  
512         <xcal:duration>  
513           <xcal:duration>PT2H</xcal:duration>  
514         </xcal:duration>  
515       </xcal:properties>  
516     </xcal:interval>  
517     <xcal:interval>  
518       <xcal:properties>  
519         <xcal:uid>  
520           <xcal:text>8edc5ef8-c269-4897-be10-  
521 c8a3eb6c8fb6</xcal:text>  
522         </xcal:uid>  
523         <xcal:related-to>  
524           <xcal:parameters>  
525             <xcal:reltype>  
526               <xcal:text>FS</xcal:text>  
527             </xcal:reltype>  
528           <xcal:gap>  
529             <xcal:duration>PT10M</xcal:duration>  
530           </xcal:gap>  
531           </xcal:parameters>  
532           <xcal:uid>0ba5a8c0-4eb2-49db-8514-  
533 5da18f53caaa</xcal:uid>  
534         </xcal:related-to>  
535         <xcal:duration>  
536           <xcal:duration>PT3H</xcal:duration>  
537         </xcal:duration>  
538       </xcal:properties>  
539     </xcal:interval>  
540   </xcal:components>  
541 </xcal:vcalendar>
```

542 In the example above, the Intervals are 1 hour, 2 hours, and 3 hours long. There is a ten minute period
543 between the second and third periods.

544 3.1.4.1 Anchoring a Sequence

545 A Sequence becomes an Anchored Sequence whenever the Designated Interval within the Sequence
546 becomes Anchored. An Interval is Anchored when it has a specific starting date and time (dtstart). A
547 Sequence may become Anchored when a Designated Interval becomes Anchored through Binding. A
548 Gluon may reference a Designated Interval through an external reference, i.e., through referring to a
549 resolvable Uid. A given Sequence may remain Unanchored while being incorporated into many Anchored
550 Sequences through multiple Gluon references each creating a different Bound dtStart.

551 *Example 3-6: An Anchored Sequence*

```
552 <xcal:vcalendar>  
553   <xcal:components>  
554     <xcal:interval>  
555       <xcal:properties>  
556         <xcal:uid>  
557           <xcal:text>86879372-6d90-4de3-8267-  
558 eae50c774f82</xcal:text>  
559         </xcal:uid>  
560         <xcal:duration>  
561           <xcal:duration>PT15M</xcal:duration>  
562         </xcal:duration>  
563       </xcal:properties>  
564     </xcal:interval>  
565     <xcal:interval>  
566       <xcal:properties>  
567         <xcal:uid>  
568           <xcal:text>c212aa90-6fc4-41e1-b8fb-  
569 a23308610247</xcal:text>  
570         </xcal:uid>  
571         <xcal:related-to>  
572           <xcal:parameters>  
573             <xcal:reltype>  
574               <xcal:text>FS</xcal:text>  
575             </xcal:reltype>  
576           </xcal:parameters>  
577           <xcal:uid>86879372-6d90-4de3-8267-  
578 eae50c774f82</xcal:uid>  
579         </xcal:related-to>  
580         <xcal:duration>  
581           <xcal:duration>PT2H</xcal:duration>  
582         </xcal:duration>  
583         <xcal:dtstart>  
584           <xcal:parameters>  
585             <xcal:tzid>  
586               <xcal:text>America/New_York</xcal:text>  
587             </xcal:tzid>  
588           </xcal:parameters>  
589           <xcal:date-time>2011-05-28T09:00:00</xcal:date-  
590 time>  
591         </xcal:dtstart>  
592       </xcal:properties>  
593     </xcal:interval>  
594     <xcal:interval>  
595       <xcal:properties>  
596         <xcal:uid>  
597           <xcal:text>50970789-1a4b-4fe5-a7e3-  
598 2da2c6db43ef</xcal:text>  
599         </xcal:uid>  
600
```



```

601         <xcal:related-to>
602             <xcal:parameters>
603                 <xcal:reltype>
604                     <xcal:text>FS</xcal:text>
605                 </xcal:reltype>
606                 <xcal:gap>
607             </xcal:parameters>
608         <xcal:duration>PT10M</xcal:duration>
609         </xcal:gap>
610     </xcal:parameters>
611     <xcal:uid>c212aa90-6fc4-41e1-b8fb-
612 a23308610247</xcal:uid>
613 </xcal:related-to>
614 <xcal:duration>
615     <xcal:duration>PT30M</xcal:duration>
616 </xcal:duration>
617 </xcal:properties>
618 </xcal:interval>
619 </xcal:components>
620 </xcal:vcalendar>

```

621 3.1.5 State Changes

622 A common service interaction is to request that, at a certain time, a discrete state change will occur. It
623 could be that the price will rise. It could be that a report will be run. Such a communication has no
624 inherent Duration.

625 While this specification extends iCalendar through the use of Intervals in Sequences, the pre-existing
626 elements of iCalendar remain in place, and more are defined periodically. State Changes can be handled
627 in one of two ways today. As iCalendar is continually extended, other ways may become available
628 tomorrow. Specifications that claim conformance to WS-Calendar SHALL state how they will
629 communicate state changes.

630 **[RFC5545]** specified the use of a VEVENT with a start date and time, but no end date and time and no
631 duration. WS-Calendar introduces the communication of state changes through use of an Interval with the
632 Duration explicitly set to zero time "P". Because the Duration is explicit, it is not be over-ridden through
633 inheritance.

634

635 *Example 3-7 State Change communication using Zero Duration Interval*

```

636 <xcal:interval>
637   <xcal:properties>
638     <xcal:uid>
639       <xcal:text>04998c75-86fb-429b-8206-0a95559feb96</xcal:text>
640     </xcal:uid>
641     <xcal:duration>
642       <xcal:duration>PT0M</xcal:duration>
643     </xcal:duration>
644     <xcal:dtstart>
645       <xcal:parameters>
646         <xcal:tzid>
647           <xcal:text>America/New_York</xcal:text>
648         </xcal:tzid>
649       </xcal:parameters>
650       <xcal:date-time>2011-05-28T16:15:00</xcal:date-time>
651     </xcal:dtstart>
652   </xcal:properties>
653 </xcal:interval>

```

654 *Example 3-8 State Change communication using Event without Duration or End*

```

655 <xcal:vevent>

```

```

656 <xcal:properties>
657   <xcal:uid>
658     <xcal:text>abfccdb0-41b5-46c5-a71d-226cea632034</xcal:text>
659   </xcal:uid>
660   <xcal:dtstart>
661     <xcal:parameters>class
662       <xcal:tzid>
663         <xcal:text>America/New_York</xcal:text>
664       </xcal:tzid>
665     </xcal:parameters>
666     <xcal:date-time>2011-05-28T16:15:00</xcal:date-time>
667   </xcal:dtstart>
668 </xcal:properties>
669 <xcal:components/>
670 </xcal:vevent>

```

671 3.2 Attachments and Tolerance

672 While iCalendar expresses time and intervals, WS-Calendar associates those intervals with specific
673 services and service performance characteristics. In iCalendar Components, the Attachment is used to
674 include information outside the scope of traditional Calendar services. WS-Calendar extends the
675 Attachment to support payloads developed in other specifications. WS-Calendar also defines a new
676 Property for iCalendar Components that specifies the Tolerance for variation in temporal performance
677 that still results in successful delivery of service.

678 3.2.1 Attachment and the Artifact

679 Each Interval contains an Attachment to provide a container for delivering a payload or for referencing an
680 external service. This payload is transported within the Interval either because it describes a service that
681 is or can be provided over an Interval, or whose service qualities vary over several Intervals in a
682 Sequence. As the Technical Committee cannot know all the specifications that may incorporate WS-
683 Calendar, this specification cannot discuss the contents of this payload. WS-Calendar does expect,
684 however, that these payloads will respect and extend the inheritance and conformance rules herein
685 specified.

686 The payload may be in-line, i.e., contained within the WS-Calendar Attach, or it may be found by
687 reference. WS-Calendar supports references either to another section of the same XML document
688 sharing the same message as WS-Calendar element, or to an external service or specification. The WS-
689 Calendar Attach can be thought of as having three options: “perform as described here”, or “perform as
690 described below”, or “perform as described elsewhere.”

691 *Table 3-3: Elements of a WS-Calendar Attachment*

Attachment Element	Discussion
Artifact	Unevaluated (by WS-Calendar) container for payload describing service.
Uri	Points to external XML, or XML located elsewhere in the document
Text	The use of text in an Attachment is allowed by but not defined in this specification.

692 Specifications that incorporate WS-Calendar may wish to restrict these choices through conformance
693 requirements.

694 *Example 3-9: Use of an Attachment with inline XML artifact*

```

695 <xcal:interval xmlns:payload="urn:not:a:real:artifact">
696   <xcal:properties>
697     <xcal:uid>
698       <xcal:text>9c829c35-061a-466e-98f5-ec1fe7b49d6a</xcal:text>

```

```

699     </xcal:uid>
700     <xcal:duration>
701         <xcal:duration>PT10H</xcal:duration>
702     </xcal:duration>
703     <xcal:x-wsCalendar-attach>
704         <payload:payload>
705             <payload:units>furlongs</payload:units>
706             <payload:quantity>11</payload:quantity>
707         </payload:payload>
708     </xcal:x-wsCalendar-attach>
709     </xcal:properties>
710 </xcal:interval>

```

711 The Artifact is any element derived from the Attach, allowing compliant XML from any namespace to be
712 submitted as a payload. As per the rules of any specification claiming conformance, the payload should
713 be Fully Bound before evaluation for completeness.

714 *Example 3-10: Use of an Attachment with external reference*

```

715 <xcal:interval>
716   <xcal:properties>
717     <xcal:uid>
718       <xcal:text>ad289a5e-44b0-4e28-9cbc-d61b715f5427</xcal:text>
719     </xcal:uid>
720     <xcal:duration>
721       <xcal:duration>PT10H</xcal:duration>
722     </xcal:duration>
723     <xcal:x-wsCalendar-attach>
724       <xcal:uri>http://examples.oasis-
725 open.org/ref/external</xcal:uri>
726     </xcal:x-wsCalendar-attach>
727   </xcal:properties>
728 </xcal:interval>

```

729 3.2.2 Tolerance: What is Timely Performance

730 The Tolerance parameter in WS-Calendar make interaction requirements explicit. In human schedule
731 interactions, different organizations have their own expectations. Meetings may start on the hour or within
732 5 minutes of the hour. As agents scheduled in those organizations, people learn the expected precision.
733 For services, that precision must be explicit to prevent interoperation problems.

734 Action coordination between systems requires precise communication about expectations for the
735 timeliness of performance. The Tolerance parameter added to any iCalendar Component makes explicit
736 the tolerance for time imprecision within a scheduled event. Tolerance can be applied to each Interval or
737 to an entire Sequence.

738 The Tolerance Property refines the meaning of time-related communication between services. All
739 elements of the Tolerance Property use the Duration element as defined in [RFC5545].

740 *Table 3-4: Tolerance Elements*

Tolerance Element	Discussion
Start Before Tolerance	Indicates how far before the requested start time the requested service may commence, for example if a service that begins at 1:57 is compliant with a request to start at 2:00
Start After Tolerance	Indicates how far after the requested start time the requested service may commence , for example, if a service that begins at 2:01 is compliant with a request to start at 2:00

Tolerance Element	Discussion
End Before Tolerance	Indicates how far before scheduled end time may end, for example, if a service that ends at 1:57 is compliant with a request to end at 2:00
End After Tolerance	Indicates how far after the scheduled end time the requested service may commence., for example, if a service that ends at 2:01 is compliant with a request to end at 2:00
Duration Long Tolerance	Indicates by how much the performance Duration may exceed the Duration specified in the information exchange. Duration Long Tolerance SHALL NOT be used when Start and End Tolerances are both specified.
Duration Short Tolerance	Indicates by how much the performance Duration may exceed the Duration specified in the information exchange. Duration Short Tolerance SHALL NOT be used when Start and End Tolerances are both specified.
Granularity	Whatever the time tolerance above, there is some minimum time that is considered significant. When used in Tolerance, Granularity defines the tracking and reporting requirements for a service.

741 Tolerance is part of the core WS-Calendar service definition. Similar products or services, identical except
742 for different Tolerance characteristics may appear in different markets. The ability to perform within
743 Tolerance influences the price offered and the service selected. Note that Tolerance parameter does not
744 indicate time, but only Duration. A Tolerance parameter associated with an Unscheduled Interval does
745 not change when that Interval is scheduled.

746 In the example, the service can start as much as 1 minute earlier than the scheduled time, and must start
747 no later than the scheduled time. Whenever the service starts, the service must execute for exactly the
748 Duration indicated.

749 Generally, the implementer should refrain from expressing unnecessary or redundant Tolerance
750 characteristics.

751 *Example 3-11: Interval with inline XML artifact and optional specified Performance*

```

752 <xcal:interval xmlns:payload="urn:not:a:real:artifact">
753   <xcal:properties>
754     <xcal:uid>
755       <xcal:text>030c603c-9e06-4dd4-8354-69dc3fa4d253</xcal:text>
756     </xcal:uid>
757     <xcal:duration>
758       <xcal:duration>PT3H30M</xcal:duration>
759     </xcal:duration>
760     <xcal:x-wsCalendar-attach>
761       <payload:payload>
762         <payload:units>furlongs</payload:units>
763         <payload:quantity>11</payload:quantity>
764       </payload:payload>
765     </xcal:x-wsCalendar-attach>
766     <xcal:tolerance>
767       <xcal:tolerate>
768         <xcal:startbefore>PT10M</xcal:startbefore>
769         <xcal:startafter>PT0M</xcal:startafter>
770       </xcal:tolerate>
771     </xcal:tolerance>
772   </xcal:properties>
773 </xcal:interval>

```

774 **3.3 Using Sequences: referencing, modifying, and remote access**

775 Sequences can define specific progressions of performance or state within a wide range of services and
776 specifications. They become more useful as they can be re-used or modified. A Sequence that is not fully
777 specified can be adapted and re-used without re-statement. An abstract Sequence can become a service
778 through iterative referencing.

779 An entire Sequence can become scheduled by scheduling a single Interval in a Sequence. A single
780 Sequence can become scheduled multiple times by repeated reference through different Gluons. The
781 terminology describing this was introduced in Table 1-6.

782 As a Sequence is reified through reference, WS-Calendar specifies how additional information is applied
783 or not applied to each Interval through a chain of references. We refer to this process as inheritance.
784 Derivative specification can take advantage of inheritance by defining specific rules that conform to the
785 WS-Calendar inheritance pattern.

786 This section describes how to create References to Sequences, including remote References, the rules
787 that allow schedule-related information to become more complete through those references, and how to
788 specify conforming rules in derivative specifications.

789 **3.3.1 References and Inheritance.**

790 Sequences are composed of Intervals for which a set of temporal relations have been defined. In WS-
791 Calendar, we refer to a Sequence by creating a Relation of type "CHILD" that references the UID of any
792 Interval in the Sequence. As defined in Table 1-6, the Interval within a Sequence that is the target of this
793 reference is the Designated Interval. The referring Component is named the Parent.

794 Wherever the Designated Interval, it can inherit that information from the referring Component. These
795 references may be local or remote. Some, but not all, of the information can be inherited by the other
796 Intervals in the Sequence.

797 Adding additional references can further specify information in the Sequence through inheritance; these
798 additional references are created by specifying an additional Gluon that has a Relation that references
799 the previous referring Component as a CHILD. In this way, we can create a grand-parent and a great
800 grand-parent.

801 A Remote Reference is a Relation to a Component external to the conveying message. A Component in
802 a message may reference a component already known to the receiving system. In this way, a remote
803 Sequence can be invoked (and scheduled) without re-definition or re-transmission.

804 Each Parent bequeaths information to its Child. A Child inherits this information in accord with the
805 inheritance rules. If the child is itself a parent, it bequeaths its information, the Bound result of its internal
806 information and its inheritance, to its child. Information to complete the specification of a Sequence flows
807 in this way from parent to child, from the outer reference to the inner Sequence.

808 Inheritance by the Designated Interval is governed by slightly different inheritance rules than the other
809 Intervals in the Sequence. In particular, only the Designated Interval can inherit the start date and time
810 from its parent. The starting date and times if other Intervals in a Sequence are computed using the
811 temporal relationships within the Sequence. Other information can be inherited by all Intervals in a
812 Sequence. The semantics used for inheritance is in *Table 1-5: Semantics: Inheritance* and conformance
813 rules for Inheritance are found in *Section 4*.

814 **3.3.1.1 Introducing the Gluon**

815 The referring Components described in 3.3.1 are named Gluons. In physics, gluons are particles that
816 affect the exchanges of force between quarks, but are not themselves quarks. By analogy, Gluons affect
817 the referencing and binding of Intervals in a Sequence, but are not themselves Intervals or part of
818 Sequences. Because Intervals can inherit almost any property from a Gluon, Gluons contain most of the
819 same information elements as Intervals. Because Intervals can contain information payloads for
820 specifications that use WS-Calendar, Gluons can contain information payloads from those specifications
821 as well.

822 Gluons reference and bind the Intervals in a Sequence, but are not themselves Intervals or part of
 823 Sequences. Gluons can contain payloads, or portions of payloads, which are not defined in this
 824 specification. Information from Gluons is inherited by their Children as described in section 4
 825 *Conformance and Rules for WS-Calendar and Referencing Specifications*.

826 The Gluon is in essence an the Interval Component profiled down to minimal elements for which
 827 inheritance rules defined, and able to carry a conforming informational payload. (See Appendix *Overview*
 828 *of WS-Calendar, its Antecedents and its Use*) Gluons use iCalendar relations to apply service information
 829 to Sequences.

830 *Table 3-5: Gluon Elements*

Gluon Element	Discussion
DtStamp	Time and date that Gluon artifact was created
Uid	Used to enable unambiguous referencing of each Gluon object
Summary	Text describing the Gluon
Child	A Gluon must have a link to at least one CHILD.
Duration	If specified, a Duration is potentially inherited by all Intervals in the referred-to Sequence.
DtStart	A Gluon may either have a dtStart or a dtEnd, but may not have both. DtStart is inherited by the Designated Interval.
DtEnd	A Gluon may have either a dtStart or a dtEnd, but may not have both. DtEnd is inherited by the Designated Interval, in which it is used with the Bound Duration to compute the Bound dtStart.
Attach	The used as a base class for extension by conforming specifications. Each contains the informational payload defined in that specification. Defined in section 3.2.
Availability	Referred to as Availability, provides information as to when a process can be scheduled.

831 It is important to distinguish between the general model of the Gluon in WS-Calendar and the more
 832 specific requirements of an incorporating specification. At its minimum, a Gluon may be only a pointer to a
 833 sequence, containing only a link to its child. A Gluon may alternately include information completing (or
 834 partially completing) the information in a Sequence; that information may vary based on what is required
 835 to make the information payload actionable within any particular transaction.

836 Because the properties of the Gluon are bequeathed to the child Sequence, they can stand for the
 837 elements in any Interval in the Sequence, as defined in the Conformance Section. An inherited element
 838 can even serve as a substitute for an Interval mandatory element. For example, Duration is mandatory for
 839 all Intervals. Intervals are able to inherit Duration from a parent. A single Duration in the Parent can be
 840 inherited by each Interval in a Sequence.

841 In this way, a Sequence in which every Interval does not have a Duration, could be made complete
 842 through inheritance. If one of those Intervals does include a Duration, the Bound Duration would be its
 843 own, rather than that it inherited from a Parent of the Sequence.

844 There is a critical distinction between an individual Gluon, which may be only a pointer to a sequence, or
 845 may have information completing (or partially completing) the information in a Sequence, and what is
 846 required to make the information payload actionable within any particular transaction.

847 3.3.1.2 Availability

848 An additional use for gluons is to expose a Sequence for remote invocation. The service offered may be
849 only sometimes available. WS-Calendar incorporates the iCalendar extension **[Vavailability]** to expose
850 this schedule.

851 **[Vavailability]** offers a means to describe recurring temporal patterns, such a weekdays from 9:00-5:00,
852 Thursday mornings until July, and thereafter Tuesday evening as well. A Vavailability component is a
853 collection of Availability components, each with its data boundaries and its recurrence patterns. The
854 parameters and properties are those defined in iCalendar, the structure is defined in the referenced
855 **[Vavailability]**, and the artifact is an optional Component of a Gluon.

856 A requestor may not be aware of all aspects of the Sequence. A service requestor does know, however,
857 the desired Start and Duration of the Designated Interval. **[Vavailability]** in a Gluon is interpreted as a
858 filter only on the Designated Interval.

859 WS-Calendar adds a single optional parameter to the **[Vavailability]** component. When a Granularity
860 component is applied, it further defines the acceptable service invocation. Granularity is discussed in the
861 next section.

862 3.3.1.3 Granularity used as part of Availability

863 Granularity can be applied both to Vavailability (the collection) and to Availability (the individual rule). If
864 Granularity is specified, then it communicates the expectation that services that invoke WS-Calendar
865 conforming services should request only Start times that match the Granularity.

866 For example, the Designated Interval of a Sequence has a Duration of One Hour, and is available on
867 weekdays from 8:30 until 11:00. Without Granularity, the Service can be Scheduled at any time that does
868 not start before 8:30, nor end after 11:00. If a Granularity of 30 minutes "PT30M" is applied, the
869 Scheduled Starts are limited to 8:30, 9:00, 8:30, and 10:00, i.e., integral multiples of the Duration of the
870 Granularity beginning at the beginning of the available window.

871 3.3.2 Gluons and Sequences

872 WS-Calendar Gluons express common service requirements for an entire Sequence. If a Gluon is parent
873 to an Interval in a Sequence, then the Gluon's Attachment expresses service attributes inheritable by all
874 Intervals in the Sequence.

875 In this example, the Sequence in the previous example is expressed using a Gluon.

876 *Example 3-12: Sequence with Performance defined in the Gluon*

```
877 <xcal:vcalendar>  
878   <xcal:components>  
879     <xcal:gluon>  
880       <xcal:properties>  
881         <xcal:uid>  
882           <xcal:text>5ffaa487-206f-46e8-b3e5-  
883 958b37477cab</xcal:text>  
884         </xcal:uid>  
885         <xcal:related-to>  
886           <xcal:parameters>  
887             <xcal:reltype>  
888               <xcal:text>CHILD</xcal:text>  
889             </xcal:reltype>  
890           </xcal:parameters>  
891           <xcal:uid>2a7de3f0-54c5-4a31-9856-  
892 6a94e6c82902</xcal:uid>  
893         </xcal:related-to>  
894         <xcal:dtstart>  
895           <xcal:parameters>  
896             <xcal:tzid>
```

```

897
898     <xcal:text>America/New_York</xcal:text>
899         </xcal:tzid>
900         </xcal:parameters>
901         <xcal:date-time>2011-05-28T08:45:00</xcal:date-
902 time>
903         </xcal:dtstart>
904         <xcal:tolerance>
905         <xcal:tolerate>
906
907     <xcal:durationlong>PT5M</xcal:durationlong>
908
909     <xcal:durationshort>PT0M</xcal:durationshort>
910         <xcal:granularity>PT5S</xcal:granularity>
911         </xcal:tolerate>
912         </xcal:tolerance>
913         <xcal:x-wsCalendar-attach>
914         <payload:payload>
915         <payload:units>furlongs</payload:units>
916         <payload:quantity>11</payload:quantity>
917         </payload:payload>
918         </xcal:x-wsCalendar-attach>
919         </xcal:properties>
920         <xcal:components/>
921     </xcal:gluon>
922     <xcal:interval>
923         <xcal:properties>
924         <xcal:uid>
925         <xcal:text>2a7de3f0-54c5-4a31-9856-
926 6a94e6c82902</xcal:text>
927         </xcal:uid>
928         </xcal:properties>
929     </xcal:interval>
930     <xcal:interval>
931         <xcal:properties>
932         <xcal:uid>
933         <xcal:text>1886ebd8-a5a9-4aa8-9b6b-
934 2869e4b711de</xcal:text>
935         </xcal:uid>
936         <xcal:related-to>
937         <xcal:parameters>
938         <xcal:reltype>
939         <xcal:text>FS</xcal:text>
940         </xcal:reltype>
941         </xcal:parameters>
942         <xcal:uid>2a7de3f0-54c5-4a31-9856-
943 6a94e6c82902</xcal:uid>
944         </xcal:related-to>
945         </xcal:properties>
946     </xcal:interval>
947     <xcal:interval>
948         <xcal:properties>
949         <xcal:uid>
950         <xcal:text>62e518b4-65df-4fea-b64a-
951 91a762ae173a</xcal:text>
952         </xcal:uid>
953         <xcal:related-to>
954         <xcal:parameters>
955         <xcal:reltype>
956         <xcal:text>FS</xcal:text>
957         </xcal:reltype>
958         </xcal:parameters>

```


959
960
961
962
963
964
965
966
967
968

```

                                <xcal:uid>1886ebd8-a5a9-4aa8-9b6b-
2869e4b711de</xcal:uid>
                                </xcal:related-to>
                                <xcal:duration>
                                    <xcal:duration>PT30M</xcal:duration>
                                </xcal:duration>
                                </xcal:properties>
                            </xcal:interval>
                    </xcal:components>
</xcal:vcalendar>

```

969 Note that the performance expectations, identical for each Interval, have moved into the Gluon. Not also
970 that while the duration for all Intervals in the partition is set in the Gluon, Interval 3 overrides that with a
971 half-hour duration assigned locally. This Gluon happens to be related to the first Interval in the Sequence;
972 there are specific use cases (discussed below) which require it to be linked to other Intervals.

973 **3.3.3 Inheritance rules for Gluons**

974 In general, the rule is that anything specified in the Parent Gluon applies to each Child. The Parent of an
975 Interval in a Sequence is parent to all Intervals in the Sequence. As a Sequence creates single temporal
976 relationship, assigning a start time (dtstart) to any Interval allows computation of the starting time for each
977 of them.

978 *Table 3-6 Gluon Inheritance rules*

Attribute	Inheritance Rules
General	A Interval or Gluon inherits its attributes through it's the 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.
Artifacts	Artifacts hold payload from other specifications. Elements within Artifacts are inherited in accord with the rules in those specifications, which must be consistent the inheritance rules in WS-Calendar. Artifacts are evaluated for completeness and conformance only after processing inheritance.
Schedule	Schedule, i.e., the start date and time, are inherited only by the Designated Interval. The start date and times of other Intervals are computed by reference to the Designated Interval. Between the Gluon bequeathing a schedule and the Designated Interval, an intervening Gluon may set Availability. It is up to the application or to the specification incorporating WS-Calendar to assert whether an Interval that is outside the Availability is conforming or not.
Availability	Availability communicates restrictions on when a service is offered. Service availability is interpreted for the Designated Interval only. If there are two Availability objects, they are evaluated for the union of the two availabilities. For example, if I am available all week from 2:00 to 6:00 in one, and available all day Tuesday in the other, then after inheritance, there remains only 2:00 to 6:00 on Tuesday,

979 3.3.4 Optimizing the expression of a Partition

980 A Partition is a set of consecutive Intervals. The expressions of a Partition can be optimized by bringing
981 the Relation and Duration into the Gluon. Notice that while the type of the relationship is defined in the
982 Gluon, the Temporal Relation for each Interval must still be expressed within the Interval.

983 *Example 3-13: Partition with Duration and Relationship defined in the Gluon*

```
984 <xcal:vcalendar>
985   <xcal:components>
986     <xcal:gluon>
987       <xcal:properties>
988         <xcal:uid>
989           <xcal:text>10795fba-5c0d-406e-b9a1-
990 6e8448d8e125</xcal:text>
991         </xcal:uid>
992         <xcal:related-to>
993           <xcal:parameters>
994             <xcal:reltype>
995               <xcal:text>CHILD</xcal:text>
996             </xcal:reltype>
997           </xcal:parameters>
998           <xcal:uid>bfe0040e-a5e0-4558-bbe1-
999 a9207004a4cc</xcal:uid>
1000         <xcal:gap>
1001           <xcal:duration>PT10M</xcal:duration>
1002         </xcal:gap>
1003       </xcal:related-to>
1004       <xcal:duration>
1005         <xcal:duration>PT50M</xcal:duration>
1006       </xcal:duration>
1007       <xcal:x-wsCalendar-attach>
1008         <payload:payload>
1009           <payload:units>students</payload:units>
1010           <payload:quantity>23</payload:quantity>
1011         </payload:payload>
1012       </xcal:x-wsCalendar-attach>
1013       <xcal:tolerance>
1014         <xcal:tolerate>
1015           <xcal:startbefore>P</xcal:startbefore>
1016           <xcal:startafter>PT5M</xcal:startafter>
1017         </xcal:tolerate>
1018       </xcal:tolerance>
1019     </xcal:properties>
1020     <xcal:components/>
1021   </xcal:gluon>
1022   <xcal:interval>
1023     <xcal:properties>
1024       <xcal:uid>
1025         <xcal:text>bfe0040e-a5e0-4558-bbe1-
1026 a9207004a4cc</xcal:text>
1027       </xcal:uid>
1028     </xcal:properties>
1029   </xcal:interval>
1030   <xcal:interval>
1031     <xcal:properties>
1032       <xcal:uid>
1033         <xcal:text>7dabb353-8d0a-44c5-a81e-
1034 e49268fd85f3</xcal:text>
1035       </xcal:uid>
1036     <xcal:related-to>
1037       <xcal:parameters>
1038         <xcal:reltype>
```

```

1039         <xcal:text/>
1040         </xcal:reltype>
1041         </xcal:parameters>
1042         <xcal:uid>bfe0040e-a5e0-4558-bbe1-
1043 a9207004a4cc</xcal:uid>
1044         </xcal:related-to>
1045         </xcal:properties>
1046     </xcal:interval>
1047     <xcal:interval>
1048         <xcal:properties>
1049             <xcal:uid>
1050                 <xcal:text>4c9ca94b-0b47-4244-b72d-
1051 e5333e1f7e43</xcal:text>
1052             </xcal:uid>
1053             <xcal:related-to>
1054                 <xcal:parameters>
1055                     <xcal:reltype>
1056                         <xcal:text/>
1057                     </xcal:reltype>
1058                 </xcal:parameters>
1059                 <xcal:uid>bfe0040e-a5e0-4558-bbe1-
1060 a9207004a4cc</xcal:uid>
1061             </xcal:related-to>
1062             </xcal:properties>
1063     </xcal:interval>
1064     <xcal:interval>
1065         <xcal:properties>
1066             <xcal:uid>
1067                 <xcal:text>397068e3-ca9a-4b3a-a1e0-
1068 76a45671bb2f</xcal:text>
1069             </xcal:uid>
1070             <xcal:related-to>
1071                 <xcal:parameters>
1072                     <xcal:reltype>
1073                         <xcal:text/>
1074                     </xcal:reltype>
1075                 </xcal:parameters>
1076                 <xcal:uid>bfe0040e-a5e0-4558-bbe1-
1077 a9207004a4cc</xcal:uid>
1078             </xcal:related-to>
1079             </xcal:properties>
1080     </xcal:interval>
1081     <xcal:interval>
1082         <xcal:properties>
1083             <xcal:uid>
1084                 <xcal:text>c97dfe10-1e2d-42c2-bdc5-
1085 41a5ca17c98a</xcal:text>
1086             </xcal:uid>
1087             <xcal:related-to>
1088                 <xcal:parameters>
1089                     <xcal:reltype>
1090                         <xcal:text/>
1091                     </xcal:reltype>
1092                 </xcal:parameters>
1093                 <xcal:uid>bfe0040e-a5e0-4558-bbe1-
1094 a9207004a4cc</xcal:uid>
1095             </xcal:related-to>
1096             </xcal:properties>
1097     </xcal:interval>
1098     <xcal:interval>
1099         <xcal:properties>
1100             <xcal:uid>

```

```

1101         <xcal:text>bf22e5e1-891f-40eb-9012-
1102 89577482f38d</xcal:text>
1103         </xcal:uid>
1104         <xcal:related-to>
1105         <xcal:parameters>
1106             <xcal:reltype>
1107                 <xcal:text/>
1108             </xcal:reltype>
1109         </xcal:parameters>
1110         <xcal:uid>bfe0040e-a5e0-4558-bbe1-
1111 a9207004a4cc</xcal:uid>
1112         </xcal:related-to>
1113     </xcal:properties>
1114 </xcal:interval>
1115 <xcal:interval>
1116     <xcal:properties>
1117         <xcal:uid>
1118             <xcal:text>2fbe48fa-3539-4167-8db6-
1119 2dae1d86f10b</xcal:text>
1120         </xcal:uid>
1121         <xcal:related-to>
1122         <xcal:parameters>
1123             <xcal:reltype>
1124                 <xcal:text/>
1125             </xcal:reltype>
1126         </xcal:parameters>
1127         <xcal:uid>bfe0040e-a5e0-4558-bbe1-
1128 a9207004a4cc</xcal:uid>
1129         </xcal:related-to>
1130     </xcal:properties>
1131 </xcal:interval>
1132 <xcal:interval>
1133     <xcal:properties>
1134         <xcal:uid>
1135             <xcal:text>5b1e382f-4ff0-43fe-9535-
1136 1d98ca2a852d</xcal:text>
1137         </xcal:uid>
1138         <xcal:related-to>
1139         <xcal:parameters>
1140             <xcal:reltype>
1141                 <xcal:text/>
1142             </xcal:reltype>
1143         </xcal:parameters>
1144         <xcal:uid>bfe0040e-a5e0-4558-bbe1-
1145 a9207004a4cc</xcal:uid>
1146         </xcal:related-to>
1147     </xcal:properties>
1148 </xcal:interval>
1149 </xcal:components>
1150 </xcal:vcalendar>

```

1151 The Partition above shows a school schedule in which classes start one hour apart. Each class is for 50
1152 minutes, and there is a 10 minute gap between each as students move between classes. Classes may
1153 not begin before the schedule, but they may start up to five minutes late.

1154 3.3.5 Notifying Partners of Process Availability

1155 A Sequence has not been scheduled until it has a start date and time. Sometimes it is useful to limit the
1156 possible start-times. For example, consider a service that is only available at 9:00 AM each day. It has not
1157 yet been scheduled, so its dtStart is empty. The Vavailability object, expressed either in the Designated
1158 Interval, or in the lineage of Gluons, is used to restrict this offering.

1159 *Example 3-14: Vavailability*

```

1160 <xcal:vavailability>
1161   <xcal:properties>
1162     <xcal:uid>
1163       <xcal:text>9ae65e93-4c68-4811-aa10-fcdbacb7ba79</xcal:text>
1164     </xcal:uid>
1165     <xcal:dtstart>
1166       <xcal:parameters>
1167         <xcal:tzid>
1168           <xcal:text>America/New_York</xcal:text>
1169         </xcal:tzid>
1170       </xcal:parameters>
1171       <xcal:date-time>2011-03-01T00:00:00</xcal:date-time>
1172     </xcal:dtstart>
1173     <xcal:dtend>
1174       <xcal:parameters>
1175         <xcal:tzid>
1176           <xcal:text>America/New_York</xcal:text>
1177         </xcal:tzid>
1178       </xcal:parameters>
1179       <xcal:date-time>2011-03-31T00:00:00</xcal:date-time>
1180     </xcal:dtend>
1181   </xcal:properties>
1182   <xcal:components>
1183     <xcal:available xs:type="xcal:AvailableType">
1184       <xcal:properties>
1185         <xcal:dtstart>
1186           <xcal:date-time>2011-03-01T09:00:00</xcal:date-
1187 time>
1188         </xcal:dtstart>
1189         <xcal:dtend>
1190           <xcal:date-time>2011-03-01T11:00:00</xcal:date-
1191 time>
1192         </xcal:dtend>
1193         <xcal:rrule>
1194           <xcal:recur>
1195             <xcal:freq>WEEKLY</xcal:freq>
1196             <xcal:byday>MO</xcal:byday>
1197             <xcal:byday>TU</xcal:byday>
1198             <xcal:byday>WE</xcal:byday>
1199             <xcal:byday>TH</xcal:byday>
1200           </xcal:recur>
1201         </xcal:rrule>
1202       </xcal:properties>
1203     </xcal:available>
1204     <xcal:available xs:type="xcal:AvailableType">
1205       <xcal:properties>
1206         <xcal:dtstart>
1207           <xcal:date-time>2011-03-01T15:00:00</xcal:date-
1208 time>
1209         </xcal:dtstart>
1210         <xcal:dtend>
1211           <xcal:date-time>2011-03-01T16:00:00</xcal:date-
1212 time>
1213         </xcal:dtend>
1214         <xcal:rrule>
1215           <xcal:recur>
1216             <xcal:freq>WEEKLY</xcal:freq>
1217             <xcal:byday>FR</xcal:byday>
1218           </xcal:recur>
1219         </xcal:rrule>
1220       </xcal:properties>

```

```

1221     </xcal:available>
1222     </xcal:components>
1223 </xcal:vavailability>

```

1224 The Vavailability above describes service availability for the month of March, 2011, i.e., it has a start date
1225 of March 1 and an end date of March 31. Within that period, there are two schedules, described by the
1226 two availability artifacts. The first specifies that starting on March 1, there is a window of 9-11 am, Eastern
1227 Time, on Monday, Tuesday, Wednesday, and Thursday each week. The second specifies another
1228 window of availability from 3:00 PM (15:00) to 4:00 PM (16:00) on Fridays. These schedules are each
1229 valid only through March 31, the dtEnd of the encompassing Vavailability. If neither date nor duration
1230 were specified, then the end of the schedules would be indefinite.

1231 The example above uses daily schedules with a weekly recurrence. The full breadth of recurrence rules is
1232 described in [iCalendar].

1233 3.3.5.1 Combining a Gluon and Availability.

1234 Consider the school schedule in the partition example in Section 3.3.4 Optimizing the expression of a
1235 Partition that is used in several examples. The school has a single valid start time, at 8:00. The service
1236 can be refined by advertising its Availability as beginning at 9:00 on the first day. Availability re-occurs on
1237 a weekly schedule, only on the weekdays Monday, Tuesday, Thursday, and Friday. Furthermore, the
1238 schedule can only be invoked during the Fall semester, from September 1, to December 15.

1239 With a Granularity of one hour set, the schedule can only begin on the time that the Availability begins, or
1240 at one hour intervals thereafter. If the Availability Window is only from 8:00 with a Duration of one hour,
1241 then the service is advertised only for a start at this hour.

1242 The example below illustrates how to use the Vavailability object contained in a gluon to publish
1243 Availability on a pre-existing sequence.

1244 *Example 3-15 Gluon publishing availability of pre-existing sequence*

```

1245 <xcal:gluon>
1246   <xcal:properties/>
1247   <xcal:components>
1248     <xcal:vavailability>
1249       <xcal:properties>
1250         <xcal:uid>
1251           <xcal:text>7cf50215-e50e-46a9-9377-
1252 797aff409ae3</xcal:text>
1253         </xcal:uid>
1254         <xcal:dtstart>
1255           <xcal:parameters>
1256             <xcal:tzid>
1257               <xcal:text>America/New_York</xcal:text>
1258             </xcal:tzid>
1259           </xcal:parameters>
1260           <xcal:date-time>2011-09-01T00:00:00</xcal:date-
1261 time>
1262         </xcal:dtstart>
1263         <xcal:dtend>
1264           <xcal:parameters>
1265             <xcal:tzid>
1266               <xcal:text>America/New_York</xcal:text>
1267             </xcal:tzid>
1268           </xcal:parameters>
1269           <xcal:date-time>2011-12-17T00:00:00</xcal:date-
1270 time>
1271         </xcal:dtend>
1272       </xcal:properties>
1273     </xcal:components>
1274   </xcal:gluon>

```

```

1276         <xcal:available xs:type="xcal:AvailableType">
1277             <xcal:properties>
1278                 <xcal:dtstart>
1279                     <xcal:date-time>2011-09-
1280 01T08:00:00</xcal:date-time>
1281                 </xcal:dtstart>
1282                 <xcal:dtend>
1283                     <xcal:date-time>2011-09-
1284 01T09:00:00</xcal:date-time>
1285                 </xcal:dtend>
1286                 <xcal:rrule>
1287                     <xcal:recur>
1288                         <xcal:freq>WEEKLY</xcal:freq>
1289                         <xcal:byday>MO</xcal:byday>
1290                         <xcal:byday>WE</xcal:byday>
1291                         <xcal:byday>FR</xcal:byday>
1292                     </xcal:recur>
1293                 </xcal:rrule>
1294             </xcal:properties>
1295         </xcal:available>
1296     </xcal:components>
1297 </xcal:vavailability>
1298 </xcal:components>
1299 </xcal:gluon>

```

1300 In the example above, the general classroom schedule has been referenced by a new gluon, and
1301 established the availability for the Fall semester. The new gluon references the pre-existing gluon that
1302 establishes the sequence as a partition.

1303 This double inheritance, in which a Sequence inherits from a Gluon which inherits from a Gluon is a
1304 useful pattern for advertising or scheduling a service.

1305 3.3.6 Other Scheduling Scenarios

1306 Sometimes, the invoker of a service is interested only in single Interval of the Sequence, but the entire
1307 Sequence is required. In the example below, the second Interval is advertised, i.e., the Gluon points to
1308 the second Interval. The first Interval might be a required ramp-period, during which the underlying
1309 process is “warming up”, and which may bring some lesser service to market during that ramp time. The
1310 ramp-down time at the end is similarly fixed. The entire Service offering is represented by the exposed (it
1311 has a public URI) Gluon.

1312 *Example 3-16: Standard Sequence with Ramp-Up and Ramp Down*

```

1313 <xcal:vcalendar>
1314     <xcal:components>
1315         <xcal:gluon>
1316             <xcal:properties>
1317                 <xcal:uid>
1318                     <xcal:text>fcd9ebc7-32f4-4a8c-9dbd-
1319 23749e6b324e</xcal:text>
1320                 </xcal:uid>
1321                 <xcal:related-to>
1322                     <xcal:parameters>
1323                         <xcal:reltype>
1324                             <xcal:text>CHILD</xcal:text>
1325                         </xcal:reltype>
1326                     </xcal:parameters>
1327                     <xcal:uid>a4cde8b9-ed43-4ca7-9eb0-
1328 5a78a02d30b8</xcal:uid>
1329                 </xcal:related-to>
1330                 <xcal:dtstart>
1331                     <xcal:parameters>

```

```

1332         <xcal:tzid>
1333
1334     <xcal:text>America/New_York</xcal:text>
1335         </xcal:tzid>
1336     </xcal:parameters>
1337     <xcal:date-time>2011-05-28T08:45:00</xcal:date-
1338 time>
1339         </xcal:dtstart>
1340     <xcal:duration>
1341         <xcal:duration>PT2H</xcal:duration>
1342     </xcal:duration>
1343     <xcal:x-wsCalendar-attach>
1344         <payload:payload>
1345             <payload:units>fortnights</payload:units>
1346             <payload:quantity>34</payload:quantity>
1347         </payload:payload>
1348     </xcal:x-wsCalendar-attach>
1349 </xcal:properties>
1350 <xcal:components/>
1351 </xcal:gluon>
1352 <xcal:interval>
1353     <xcal:properties>
1354         <xcal:uid>
1355             <xcal:text>c5473482-2ecb-4a9b-b53c-
1356 91ff5614dc40</xcal:text>
1357         </xcal:uid>
1358         <xcal:duration>
1359             <xcal:duration>PT10M</xcal:duration>
1360         </xcal:duration>
1361     </xcal:properties>
1362 </xcal:interval>
1363 <xcal:interval>
1364     <xcal:properties>
1365         <xcal:uid>
1366             <xcal:text>a4cde8b9-ed43-4ca7-9eb0-
1367 5a78a02d30b8</xcal:text>
1368         </xcal:uid>
1369         <xcal:related-to>
1370             <xcal:parameters>
1371                 <xcal:reltype>
1372                     <xcal:text>FS</xcal:text>
1373                 </xcal:reltype>
1374             </xcal:parameters>
1375             <xcal:uid>c5473482-2ecb-4a9b-b53c-
1376 91ff5614dc40</xcal:uid>
1377         </xcal:related-to>
1378         <xcal:x-wsCalendar-attach>
1379             <payload:payload>
1380                 <payload:units>furlongs</payload:units>
1381                 <payload:quantity>11</payload:quantity>
1382             </payload:payload>
1383         </xcal:x-wsCalendar-attach>
1384     </xcal:properties>
1385 </xcal:interval>
1386 <xcal:interval>
1387     <xcal:properties>
1388         <xcal:uid>
1389             <xcal:text>de2aa95b-f930-44d5-bc2b-
1390 1356f8732879</xcal:text>
1391         </xcal:uid>
1392         <xcal:related-to>
1393             <xcal:parameters>
1394                 <xcal:reltype>

```


1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413

```

                    <xcal:text>FS</xcal:text>
                </xcal:reltype>
            </xcal:parameters>
            <xcal:uid>a4cde8b9-ed43-4ca7-9eb0-
5a78a02d30b8</xcal:uid>
        </xcal:related-to>
        <xcal:duration>
            <xcal:duration>PT5M</xcal:duration>
        </xcal:duration>
        <xcal:x-wsCalendar-attach>
            <payload:payload>
                <payload:units>furlongs</payload:units>
                <payload:quantity>11</payload:quantity>
            </payload:payload>
        </xcal:x-wsCalendar-attach>
    </xcal:properties>
</xcal:interval>
</xcal:components>
</xcal:vcalendar>

```

1414
1415
1416

The underlying sequence has a fixed warm up and cool down (intervals 1 and 3). The Gluon shares a payload with Interval 2, which has no duration. Interval 2 inherits the quantity (14) and the duration (2H) from the Gluon.

1417
1418
1419
1420

If expressed all at once, the Gluon merely provides a handle for the Sequence. A more useful expression would have the Gluon separate, or perhaps inheriting its information from a market agreement. This enables the service interaction to express that Start Time, Duration and Quantity. All three are inherited, in this case, only by the Designated Interval.

1421

3.4 Time Stamps

1422
1423
1424

Time stamps are used everywhere in inter-domain service performance analysis and have particular use to support event forensics. Time stamps may be assembled and collated from events across multiple time zones and from multiple systems.

1425
1426
1427
1428

Different systems may track time and therefore record events with different levels of Tolerance. It is not unusual for a time-stamped event from a domain with low Tolerance to appear to have occurred after one or more time-stamped events from a domain with high Tolerance. A fully qualified time-stamp includes the Granularity measure.

1429

Table 3-7: Elements of Time Stamps

Time Stamp Element	Note (Non-Normative)
Time Stamp	Fully qualified date and time of event..
Accuracy	Identifies whether an interval of a particular duration is indeed an interval of the mentioned duration plus or minus some number of milliseconds, seconds and minutes.
Time Stamp Realm	Identifies the system where the TimeStamp value originated. A set of recordings 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.

Time Stamp Element	Note (Non-Normative)
Leap Seconds Known	Indicates that the time source of the sending device support leap seconds adjustments.
Clock Failure	Indicates that the time source of the sending device is unreliable. This may put in doubt the advisability of direct comparison of this timestamp information from this system and from foreign systems,
Clock Not Synchronized	Indicates that the time source of the sending device is not synchronized with the external UTC time source.
Time Source Accuracy	Represents the time accuracy class of the time source of the sending device relative to the external UTC time source.

1430 **3.4.1 Time Stamp Realm Discussion**

1431 Within a single system, or synchronized system of systems, one can sort the temporal order of event by
 1432 sorting them by TimeStamp. Determining the order of events is the first step of event forensics. This
 1433 assumption does not apply when events are gathered across systems.

1434 Different systems may not have synchronized time, or may synchronize time against different sources.
 1435 This means different system clocks may drift apart. It may be that a later timestamp from one system
 1436 occurred before an earlier timestamp in another. As this drift is unknown, it cannot be automatically
 1437 corrected for without additional information.

1438 The TimeStampRealm element identifies which system created an event time stamp. The
 1439 TimeStampRealm identifies a source system in inter-domain interactions (a system of systems). For
 1440 example: <http://SystemA.com> and <http://SystemB.com> identify 2 systems. This example assumes
 1441 SystemA and SystemB do not have a common time source.

1442 The TimeStampRealm can also be used to identify sub-systems in intra-domain interactions (sub-systems
 1443 of a system). For example: <http://SystemA.com/SubSystem1> and <http://SystemA.com/SubSystem2>
 1444 identify 2 subsystems of the same higher level system. In cases where the upper level SystemA does not
 1445 have a global time source for synchronizing all of its sub-systems, it can be useful to identify sub-systems
 1446 in this manner.

1447 **4 Conformance and Rules for WS-Calendar and**
1448 **Referencing Specifications**

1449 **4.1 Introduction**

1450 This section specifies conformance related to the information model. If the implementer is merely using
1451 WS-Calendar as part of a larger business or service communication, they SHALL follow not only the
1452 semantic rules herein, but SHALL also conform to the rules for specifying inheritance in referencing
1453 standards.

1454 **4.2 Conformance Rules for WS-Calendar**

1455 There are five kinds of conformance that must be addressed for WS-Calendar and specifications that
1456 reference WS-Calendar.

- 1457
- 1458 • Conformance to the **inheritance rules** in WS-Calendar, including the direction of inheritance
 - 1459 • **Specific attributes** for each type that MUST or MUST NOT be inherited
 - 1460 • **Conformance rules** that Referencing Specifications MUST follow
 - 1461 • Description of **Covarying attributes** with respect to the Reference Specification
 - 1462 • **Semantic Conformance** for the information within the artifacts exchanged

1462 We address each of these in the following sections

1463 **4.2.1 Inheritance in WS-Calendar**

1464 In this section we define rules that define inheritance including direction.

1465 **I1: Proximity Rule** Within a given lineage, inheritance is evaluated through each Parent to the Child
1466 before what the Child bequeaths is evaluated.

1467 **I2: Direction Rule** Intervals MAY inherit attributes from the nearest gluon subject to the Proximity Rule
1468 and Override Rule, provided those attributes are defined as Inheritable.

1469 **I3: Override Rule** If and only if there is no value for a given attribute of a Gluon or Interval, that Gluon or
1470 Interval SHALL inherit the value for that attribute from its nearest Ancestor in conformance to the
1471 Proximity Rule.

1472 **I4: Comparison Rule** Two Sequences are equivalent if a comparison of the respective Intervals
1473 succeeds as if each Sequence were fully Bound and redundant Gluons are removed.

1474 **I5: Designated Interval Inheritance** [To facilitate composition of Sequences] the Designated Interval in
1475 the ultimate Ancestor of a Gluon is the Designated Interval of the composed Sequence. Special
1476 conformance rules for Designated Intervals apply only to the Interval linked from the Designator Gluon.

1477 **I6: Start Time Inheritance** When a start time is specified through inheritance, that start time is inherited
1478 only by the Designated Interval; the start time of all other Intervals are computed through the durations
1479 and temporal relationships within the Sequence. The Designated Interval is the Interval whose parent is
1480 at the end of the lineage.

1481 **4.2.2 Specific Attribute Inheritance**

1482 In WS-Calendar the following attributes MUST be inherited in conformance to the Rules (same for Gluons
1483 and Intervals):

- 1484
- dtStart

- 1485 • dtEnd
 - 1486 • Duration
 - 1487 • Designated Interval (Gluon, special upward inheritance rule)
 - 1488 • Tolerance
- 1489 In WS-Calendar the following attributes MUST NOT be inherited
- 1490 • UID (Gluons and Intervals)
 - 1491 • Temporal Relationships (between Intervals)

1492 4.2.3 General Conformance Issues

1493 This specification is general purpose. Standards that claim conformance to this specification may need to
 1494 restrict the variability inherent in the expressions of Date and Time to improve interoperability within their
 1495 own interactions. Aspects of Date and Time that may reward attention and conformance statements
 1496 include:

- 1497 • **Precision** – Does the conforming specification express time in Hours or in milliseconds. Consider
 1498 a standard format recommendation.
- 1499 • **Time Zones and UTC** – Business interactions have a “natural” choice of local, time zone, or UTC
 1500 based expression of time. Intents may be local, as they tie to the business processes that drive
 1501 them. Tenders may be Time-zone based, as they are driven by the local business process, but
 1502 may require future action across changes in time and in time zone. Transaction recording may
 1503 demand UTC, for complete unambiguity. The specification cannot require one or another, but
 1504 particular business processes may require appropriate conformance statements.
- 1505 • **Business Purpose** – Because WS-Calendar is general purpose, it does not distinguish between
 1506 different exchanges that may have different purposes. For example, a general indication of
 1507 capability and/or timeliness may be appropriate for a market tender, and an unanchored
 1508 Sequence may be appropriate. In the same specification, performance execution could require
 1509 merely the Gluon to Anchor the Interval. If the distinction between Unanchored and Anchored
 1510 Interval is critical for a set of interactions, the referencing specification SHALL indicate the proper
 1511 form for a given exchange.

1512 4.2.4 Covarying Elements

1513 Some elements of WS-Calendar objects may be **covarying**, meaning that they change together. Such
 1514 elements are treated as a single element for inheritance, they are either inherited together or the child
 1515 keeps its current values intact. This becomes important if one or more of a covarying set have default
 1516 values. In that case, if any are present, then inheritance should deem they are all present, albeit some
 1517 perhaps in their default values.

1518 4.2.5 Conformance of Intervals

1519 4.2.5.1 Intervals

- 1520 WS-Calendar Intervals SHALL have a Duration.
- 1521 Intervals MAY have a Start Time.
- 1522 Intervals SHALL have a Duration AND a dtStart OR a dtEnd. If a non-compliant Interval is received with
 1523 both a dtStart and a dtEnd, then the dtEnd SHALL be ignored.
- 1524 Within a Sequence, a maximum of a single Interval MAY have a dtStart or a dtEnd.

1525 4.2.5.2 Other Elements

1526 A Tolerance Property component SHALL NOT include Start, Stop, and Duration elements. Two out of the
 1527 three elements is acceptable, but not three.

1528 In Partitions, the Description, Summary and Priority of each Interval SHALL be excluded.

1529 A Gluon may have either a dtStart or a dtEnd, but may not have both.

1530 **4.2.6 Conformance of Bound Intervals and Sequences**

1531 Actionable services require Bound Intervals as part of a Bound Sequence. Services may include Intervals
1532 that are not bound for informational or negotiation purposes. Some of these are modeled and described
1533 as constraints in the UML models that have been produced separately.

- 1534 • Intervals SHALL have values assigned for dtStart and duration, either explicitly or through
1535 inheritance
- 1536 • Intervals SHALL have no value assigned for dtEnd
- 1537 • Within a Sequence at most the Designated Interval may have dtStart and duration with a value
1538 specified or inherited.
- 1539 • If Sequences are composed to create other Sequences, then the Designated Intervals within the
1540 composing Sequence are ignored.
- 1541 • Any specification claiming conformance to WS-Calendar MUST satisfy all of the following
1542 conditions:
 - 1543 ○ Follow the same style of inheritance (per the Rules)
 - 1544 ○ Specify attribute inheritability in the specification claiming conformance
 - 1545 ○ Specify whether certain sets of elements must be inherited as a group or specify that all
1546 elements can be inherited or not on an individual basis

1547 **4.3 Conformance Rules for Specifications Claiming Conformance to** 1548 **WS-Calendar**

1549 Specifications that claim conformance to WS-Calendar SHALL specify inheritance rules for use within
1550 their specification. These rules SHALL NOT violate or override the Proximity, Direction, or Override Rules. If
1551 the specification includes covariant elements, those elements SHALL be clearly designated in the
1552 specification.

1553 Specifications that normatively reference and claim conformance with WS-Calendar SHALL define the
1554 business meaning of zero duration Intervals.

1555 **4.4 Security Considerations**

1556 Part 1 of WS-Calendar describes an informational model. Specifications claiming conformance with WS-
1557 Calendar may use the schedule and interval communication as but a small part of their overall
1558 communications.

1559 Communications that claim conformance to this specification should select the communication and the
1560 well-known methods to secure that communication appropriate to the information exchanged and paying
1561 heed to the costs of both communication failure and of inappropriate disclosure. To the extent that normal
1562 schedule servers are used, the capabilities of security of those systems should be considered as well.
1563 Those concerns are out of scope for this specification.

1564 Specifications which do not use the REST or SOAP interactions face similar concerns in designing the
1565 authentication, authorization, interactions, and storage of the information artifacts produced. Such
1566 concerns are out of scope within this general model.

1567

Acknowledgements

1568 The following individuals have participated in the creation of this specification and are gratefully
1569 acknowledged:

1570 **Participants:**

1571 Bruce Bartell, Southern California Edison
1572 Brad Benson, Trane
1573 Edward Cazalet, Individual
1574 Toby Considine, University of North Carolina at Chapel Hill
1575 William Cox, Individual
1576 Sharon Dinges, Trane
1577 Mike, Douglass, Rensselaer Polytechnic Institute
1578 Craig Gemmill, Tridium, Inc.
1579 Girish Ghatikar, Lawrence Berkeley National Laboratory
1580 Gerald Gray, Southern California Edison
1581 David Hardin, ENERNOC
1582 Gale Horst, Electric Power Research Institute (EPRI)
1583 Gershon Janssen, Individual
1584 Ed Koch, Akuacom Inc.
1585 Benoit Lepeuple, LonMark International*
1586 Carl Mattocks, CheckMi*
1587 Robert Old, Siemens AG
1588 Alexander Papaspyrou, Technische Universitat Dortmund
1589 Joshua Phillips, ISO/RTO Council (IRC)
1590 Jeremy J. Roberts, LonMark International
1591 David Thewlis, CalConnect

1592
1593 The Calendaring and Scheduling Consortium (CalConnect) TC-XML committee worked closely with WS-
1594 Calendar Technical Committee, bridging to developing IETF standards and contributing the services
1595 definitions that make up Services in Section 4. The Technical Committee gratefully acknowledges their
1596 assistance and cooperation as well. Contributors to TC XML include:

1597 Cyrus Daboo, Apple
1598 Mike Douglass, Rensselaer Polytechnic Institute
1599 Steven Lees, Microsoft
1600 Tong Li, IBM

1601

1602

1603

An Introduction to Internet Calendaring

1604 *The WS-Calendar Technical Committee thanks CalConnect for contributing this overview of iCalendar*
1605 *and its use.*

1606 icalendar

1607 History

1608 The iCalendar specification was first produced by the IETF in 1998 as RFC 2445 [1]. Since then it has
1609 become the dominant standard for calendar data interchange on the internet and between devices
1610 (desktop computers, mobile phones etc.). The specification was revised in 2009 as RFC 5545 [4].

1611 Alongside iCalendar is the iTIP specification (RFC 2446 [2] and revised as RFC 5546[5]) that defines how
1612 iCalendar is used to carry out scheduling operations (for example, how an organizer can invite attendees
1613 to a meeting and receive their replies). This forms the basis for email-based scheduling using iMIP (the
1614 specification that describes how to use iTIP with email - RFC 6047 [3]).

1615 iCalendar itself is a text-based data format. However, an XML format is also available, providing a one-to-
1616 one mapping to the text format (draft [7]).

1617 iCalendar data files typically have a .ics file name extension. Most desktop calendar clients can import or
1618 export iCalendar data, or directly access such data over the Internet using a variety of protocols.

1619 Data model

1620 The iCalendar data format has a well defined data model. "iCalendar objects" encompass a set of
1621 "iCalendar Components" each of which contains a set of "iCalendar properties" and possibly other sub-
1622 Components. An iCalendar property consists of a name, a set of optional parameters (specified as "key-
1623 value" pairs) and a value.

1624 iCalendar Components include:

1625 "VEVENT" which represents an event

1626 "VTODO" which represents a task or to-do

1627 "VJOURNAL" which represents a journal entry

1628 "VFREEBUSY" which represents periods of free or busy time information

1629 "VTIMEZONE" which represents a timezone definition (timezone offset and daylight saving rules)

1630 "VALARM" is currently the only defined sub-Component and is used to set alarms or reminders on events
1631 or tasks.

1632 Properties include:

1633 "DTSTART" which represents a start time for a Component

1634 "DTEND" which represents an end time for a Component

1635 "SUMMARY" which represents a title or summary for a Component

1636 "RRULE" which can specify rules for repeating events or tasks (for example, every day, every week on
1637 Tuesdays, etc.)

1638 "ORGANIZER" which represents the calendar user who is organizing an event or assigning a task

1639 "ATTENDEE" which represents calendar users attending an event or assigned a task

1640 In addition to this data model and the pre-defined properties, the specification defines how all those are
1641 used together to define the semantics of calendar objects and scheduling. The semantics are basically a
1642 set of rules stating how all the Components and properties are used together to ensure that all iCalendar
1643 products can work together to achieve good interoperability. For example, a rule requires that all events

1644 must have one and only one "DTSTART" property. The most important part of the iCalendar specification
1645 is the semantics of the calendaring model that it represents. The use of text or XML to encode those is
1646 secondary.

1647 **Scheduling**

1648 The iTIP specification defines how iCalendar objects are exchanged in order to accomplish the key task
1649 needed to schedule events or tasks. An example of a simple workflow is as follows:

- 1650 1. To schedule an event, an organizer creates the iCalendar object representing the event and adds
1651 calendar users as attendees.
- 1652 2. The organizer then sends an iTIP "REQUEST" message to all the attendees.
- 1653 3. Upon receipt of the scheduling message, each attendee can decide whether they want to attend
1654 the meeting or not.
- 1655 4. Each attendee can then respond back to the organizer using an iTIP "REPLY" message
1656 indicating their own attendance status.

1657 iTIP supports other types of scheduling messages, for example, to cancel meetings, add new instances to
1658 a repeating meeting, etc.

1659 **Extensibility**

1660 iCalendar was designed to be extensible, allowing for new Components, properties and parameters to be
1661 defined as needed. A registry exists to maintain the list of standard extensions with references to their
1662 definitions to ensure anyone can use them and work well with others.

1663 **Calendar data access and exchange protocols**

1664 **Internet Calendar Subscriptions**

1665 An Internet calendar subscription is simply an iCalendar data file made available on a web server. Users
1666 can use this data in two ways:

1667 The data can be downloaded from the web server and then imported directly into an iCalendar aware
1668 client. This solution works well for calendar data that is not likely to change over time (for example
1669 the list of national holidays for the next year).

1670 Calendar clients that support "direct" subscriptions can use the URL to the calendar data on the web
1671 server to download the calendar data themselves. Additionally, the clients can check the web
1672 server on a regular basis for updates to the calendar data, and then update their own cached
1673 copy of it. This allows calendar data that changes over time to be kept synchronized.

1674 **CalDAV**

1675 CalDAV is a calendar access protocol and is defined in RFC 4791 [6]. The protocol is based on WebDAV
1676 which is an extension to HTTP that provides enhanced capabilities for document management on web
1677 servers.

1678 CalDAV is used in a variety of different environments, ranging from very large internet service providers,
1679 to large and small corporations or institutions, and to small businesses and individuals.

1680 CalDAV clients include desktop applications, mobile devices and browser-based solutions. It can also be
1681 used by "applets", for example, a web page panel that displays a user's upcoming events.

1682 One of the key aspects of CalDAV is its data model. Simply put, it defines a "calendar home" for each
1683 calendar user, within which any number of "calendars" can be created. Each "calendar" can contain any
1684 number of iCalendar objects representing individual events, tasks or journal entries. This data model
1685 ensures that clients and servers can interoperate well.

1686 In addition to providing simple operations to read, write and delete calendar data, CalDAV provides a
1687 querying mechanism to allow clients to fetch calendar data matching specific criteria. This is commonly
1688 used by clients to do "time-range" queries, i.e., find the set of events that occur within a given start/end
1689 time period.

1690 CalDAV also supports access control allowing for features such as delegated calendars and calendar
1691 sharing.

1692 CalDAV also specifies how scheduling operations can be done using the protocol. Whilst it uses the
1693 semantics of the iTIP protocol, it simplifies the process by allowing simple calendar data write operations
1694 to trigger the sending of scheduling messages, and it has the server automatically process the receipt of
1695 scheduling messages. Scheduling can be done with other users on the CalDAV server or with calendar
1696 users on other systems (via some form of "gateway").

1697 **ActiveSync/SyncML**

1698 ActiveSync and SyncML are technologies that allow multiple devices to synchronize data with a server,
1699 with calendar data being one of the classes of data supported. These have typically been used for low-
1700 end and high-end mobile devices.

1701 **CalWS**

1702 CalWS is a web services calendar access API developed by The Calendaring and Scheduling
1703 Consortium and the OASIS organization, to be used as part of the Oasis WS-Calendar standard. It
1704 provides an API to access and manipulate calendar data stored on a server. It follows a similar data
1705 model to CalDAV and has been designed to co-exist with a CalDAV service offering the same data.

1706 **iSchedule**

1707 iSchedule is a protocol to allow scheduling between users on different calendaring systems and across
1708 different internet domains. It transports iTIP scheduling messages using HTTP between servers. Servers
1709 use DNS and various security mechanisms to determine the authenticity of messages received.

1710 It has been specifically designed to be independent of any calendar system in use at the endpoints, so
1711 that it is compatible with many different systems. This allows organizations with different calendar
1712 systems to exchange scheduling messages with each other, and also allows a single organization with
1713 multiple calendar systems (for example due to mergers, or different departmental requirements) to
1714 exchange scheduling messages between users of each system.

1715 **References**

1716 [1] <https://datatracker.ietf.org/doc/rfc2445/> : 'Internet Calendaring and Scheduling Core Object
1717 Specification'

1718 [2] <https://datatracker.ietf.org/doc/rfc2446/> : 'iCalendar Transport-Independent Interoperability Protocol'

1719 [3] <https://datatracker.ietf.org/doc/rfc6047/> : 'iCalendar Message-Based Interoperability Protocol'

1720 [4] <https://datatracker.ietf.org/doc/rfc5545/> : 'Internet Calendaring and Scheduling Core Object
1721 Specification'

1722 [5] <https://datatracker.ietf.org/doc/rfc5546/> : 'iCalendar Transport-Independent Interoperability Protocol'

1723 [6] <https://datatracker.ietf.org/doc/rfc4791/> : 'Calendaring Extensions to WebDAV'

1724 [7] <https://datatracker.ietf.org/doc/draft-daboo-et-al-icalendar-in-xml/> : 'xCal: The XML format for
1725 iCalendar'

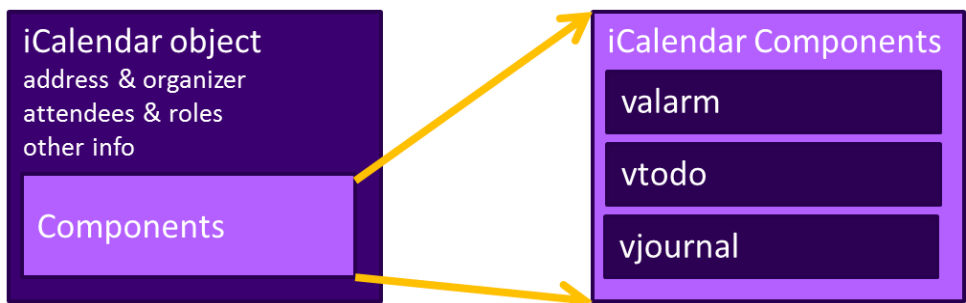
1726

1727 **Overview of WS-Calendar, its Antecedents and its Use**

1728 iCalendar has long been the predominant message format for an Internet user to send meeting requests
 1729 and tasks to other Internet users by email. The recipient can respond to the sender easily or counter
 1730 propose another meeting date/time. iCalendar support is built into all major email systems and email
 1731 clients. While SMTP is the predominant means to transport iCalendar messages, protocols including
 1732 WebDAV and SyncML are used to transport collections of iCalendar information. No similar standard for
 1733 service interactions has achieved similar widespread use.

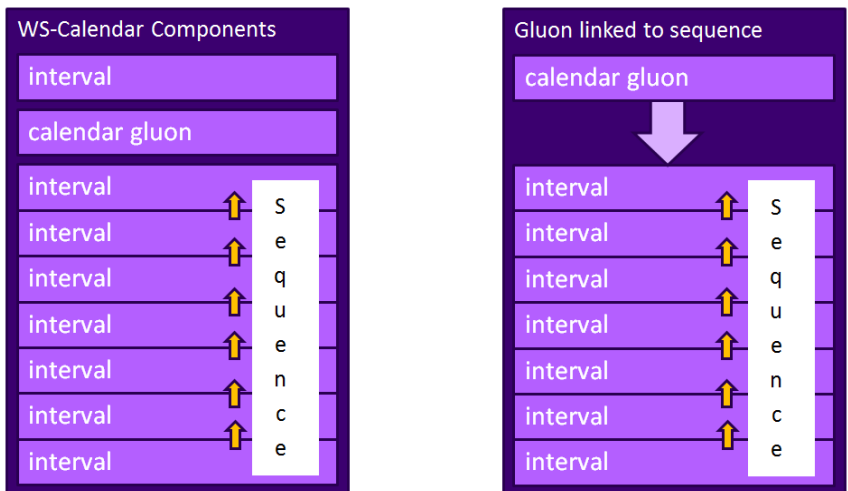
1734 The Calendar and Scheduling Consortium (CalConnect), working within the IETF, updated the iCalendar
 1735 standard in the summer of 2009 to support extension ([RFC5545]). In 2010, the same group defined
 1736 [XCAL], a canonical XML serialization for iCalendar, currently (08/21/2008) on the recommended
 1737 standards track within the IETF. This specification supports extensions, including handling non-standard,
 1738 i.e., non-iCalendar, data during message storage and retrieval.

1739 WS-Calendar builds on this work, and consists of extensions to the vocabulary of iCalendar, along with
 1740 standard services to extend calendaring and scheduling into service interactions. iCalendar consists of a
 1741 number of fields that support the delivery, update, and synchronization of if calendar messages and a list
 1742 of Components. The Components can specify defined relationships between each other.



1743
 1744 *Figure 1: iCalendar overview*

1745 WS-Calendar defines the Interval, a profile of the VTODDO Component requiring only a duration and an
 1746 artifact to define service delivery and performance. WS-Calendar also defines the CalendarGluon
 1747 Component, a container for holding only a service delivery and performance artifact, to associate with a
 1748 Component or group of Components.



1749
 1750 *Figure 2: WS-Calendar and EMIX*

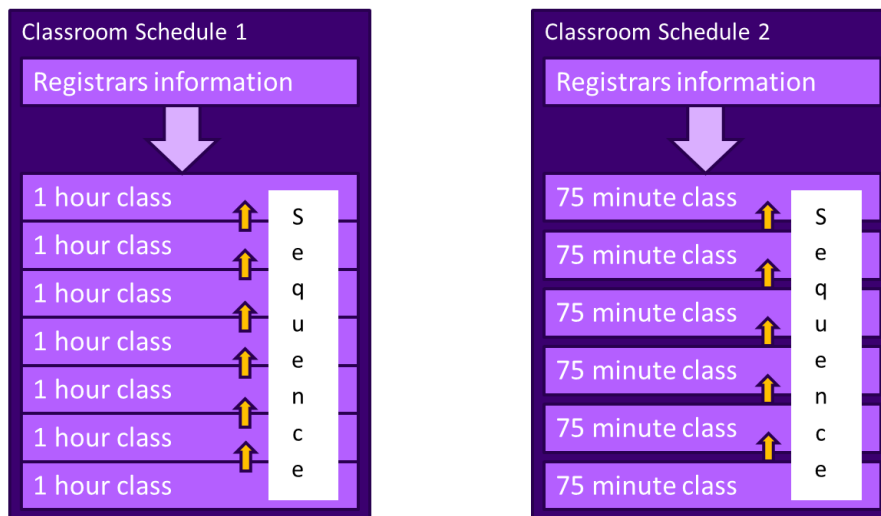
1751 A set of Intervals that have defined temporal relationships is a Sequence. Temporal relationships express
1752 how the occurrence of one Interval is related to another. For example, Interval B may begin 10 minutes
1753 after Interval A completes, or Interval D may start 5 minutes after Interval C starts. A Gluon linked to a
1754 Sequence defines service performance for all Intervals in the Sequence. Because each Interval has its
1755 own service performance contract, specifications built on WS-Calendar can define rules for inheritance
1756 and over-rides with a Sequence.

1757 The Partition is a sub-class of a Sequence in which all Intervals follow consecutively with no lag time.
1758 Intervals in a Partition normally have the same Duration, but WS-Calendar does support overriding the
1759 duration on an individual basis.

1760 Scheduling Sequences

1761 A Sequence is a general pattern of behaviors and results that does not require a specific schedule. A
1762 publishing service may advertise a Sequence with no schedule, i.e., no specific time for performance.
1763 When the Sequence is invoked or contracted, a specific performance time is added. In the original
1764 iCalendar Components, this would add the starting date and time (dtStart) to the Component. In WS-
1765 Calendar, we add the starting date and time only to the first Interval of a Sequence; the performance
1766 times for all other Intervals in the Sequence are derived from that one start time.

1767 Academic Scheduling example



1768
1769 *Figure 3: Classroom Scheduling Example*

1770 A college campus uses two schedules to schedule its buildings. In Schedule 1, classes start on the hour,
1771 and follow one after another; each class starts on the hour. In the second schedule, each class lasts an
1772 hour and a quarter, and there is a fifteen minute gap between classes; classes start on the half hour. On
1773 many campuses, the Sequence in Schedule 1 may describe classes taught on Monday, Wednesday, and
1774 Friday. Schedule 2 may describe classes taught on Tuesday and Thursday.

1775 The registrar's office knows some key facts about each classroom, including whether it hosts a class
1776 during a particular period, and the number of students that will be in that class. The college wishes to
1777 optimize the provision of building services for each class. Such services may include adequate ventilation
1778 and comfortable temperatures to assure alert students. Other services may ensure that the classroom
1779 projection systems and A/V support services are warmed up in advance of a class, or powered off when a
1780 classroom is vacant.

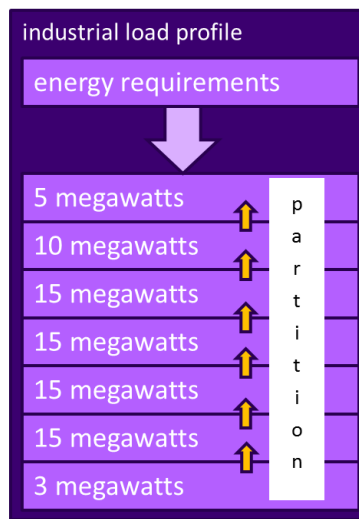
1781 Although most classes meet over typical schedule for the week (M-W-F or Tu-Th), some classes may not
1782 meet on Friday, or may have a tutorial section one day a week. The registrar's system, ever mindful of
1783 student privacy, shares only minimal information with the building systems such as how many students
1784 will be supported.

1785 The Registrar's system schedule building systems using the Gluon (registrar's information) and the
1786 student counts for each Interval, and schedules the Sequence in classroom schedule 1 three days a
1787 week for the next 10 weeks. The Registrar's system also schedules the Sequence in classroom schedule
1788 2 two days a week, also for 10 weeks.

1789 This example demonstrates a system (A) that offers services using either of two Sequences. Another
1790 business system (B) with minimal knowledge of how (A) works determines the performance requirements
1791 for (A). The business system (B) communicates what these expectations are by scheduling the
1792 Sequences offered by (A).

1793 Market Performance schedule

1794 A factory relies on an energy-intensive process which is performs twice a year for eight weeks. The
1795 factory has some flexibility about scheduling the process; it can perform the work in either the early
1796 morning or the early evening; it avoids the afternoon when energy costs are highest. The factory works up
1797 a detailed profile of when it will need energy to support this process.



1798
1799 *Figure 4: Daily Load Profile for Market Operations Example*

1800 Factory management has decided that they want to use only renewable energy products for this process.
1801 They approach two regional wind farms with the intent of making committed purchases of wind energy.
1802 The wind farms consider their proposals taking into account the seasonal weather forecasts they use to
1803 project their weather capacity, and considering the costs that may be required to buy additional wind
1804 energy on the spot market to make up any shortfalls.

1805 Each energy supplier submits of the same Sequence, a schedule, i.e. a daily starting time, and a price for
1806 the season's production. After considering the bids, and other internal costs of each proposal, the factory
1807 opts to accept a contract for the purchase of a fixed load profile (Partition), using the evening wind
1808 generation from one of the suppliers. This contract specifies Schedules of load purchases (starting data
1809 and time for the Sequence) for each day.

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.

1.0 WD 13		Toby Considine	Mechanistic processing of trivial comments for grammar, spelling, etc.
1.0 WD 14	2011-01-17	Toby Considine	Added Conformance rules, redefined inheritance, added terminology section in Section 1, added language on separability of information model, REST, and SOAP sections
1.0 WD 15	2011-01-27	Toby Considine	Pulled more definitions into Terminology Section, re-factored into multiple tables, Added Availability. Have not updated examples.
1.0 WD 15	2011-01-29	Toby Considine	Re-added footers to document (!!!) Added disclaimers on completeness prior to committee spec draft.
1.0 WD16	2011-02-07	Toby Considine	Minor changes to prepare for CSD as directed by TC
1.0 WD17	2011-03-01	Toby Considine	Reworked all examples, responded to numerous Jira editorial comments, eliminated "Mixed Inheritance of Schedule", introduced Vavailability, eliminated UML chapter which confused more than enlightened.
1.0 WD18	2011-03-16	Toby Considine William Cox	Tightened language, spelling and grammar, consolidated chapters into "larger sections" Corrected to use CHILD link instead of PARENT in conformance with RFC5545. Replaced LINK language that was leftover from earlier schemas.
1.0 WD19	2011-03-19	Toby Considine	Changes to namespace to prepare for CSD, PR02, as directed by TC vote on 3/18/2011
1.0 WD20	2011-05-12	Toby Considine	Mechanical edits. Rebuilt document to remove cross-reference corruption (table and example lists), applied grammatical and punctuation changes from PR02, simple global replaces of terms. Reference checks. Refinement of logic of Duration/DtStart. Eliminated redefinition of VAVAILABILITY.
1.0 WD21	2011-05-16	Toby Considine	More Jira edits, especially unscrambling dtStart, dtEnd, and Duration, Vavailability, and many reference checks.
1.0 WD22	2011-05-20	William Cox	Eliminated Parts two and Three
1.0 WD23	2011-05-23	Toby Considine	Minor edits for clarity, final Jira issues
1.0 WD24	2011-05-26	Toby Considine	Examples updated
1.0 WD25	2011-05-26	William Cox	Eliminated remaining references to Parts Two and Three, corrected internal links