WS-Calendar Version 1.0

Working Draft 11

11 September 2010

Specification URIs:
This Version:
http://docs.oasis-open.org/WS-Calendar/v1.0/wd11/WS-Calendar-1.0-spec-wd-11.pdf
http://docs.oasis-open.org/WS-Calendar/v1.0/wd11/WS-Calendar-1.0-spec-wd-11.html
http://docs.oasis-open.org/WS-Calendar/v1.0/wd11/WS-Calendar-1.0-spec-wd-11.doc
Previous Version:

N/A

Latest Version:
http://docs.oasis-open.org/WS-Calendar/v1.0/WS-Calendar-1.0-spec.pdf
http://docs.oasis-open.org/WS-Calendar/v1.0/WS-Calendar-1.0-spec.html
http://docs.oasis-open.org/WS-Calendar/v1.0/WS-Calendar-1.0-spec.doc
Technical Committee:
OASIS WS-Calendar TC
Chair(s):
Toby Considine

Editor(s):
Toby Considine

Related work:

This specification replaces or supersedes:

N/A

This specification is related to:

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

Declared XML Namespace(s):

http://docs.oasis-open.org/ns/WS-Calendar/WS-Calendar-201001

Abstract:

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

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

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

Status:

This document was last revised or approved by the WS-Calendar Technical Committee on the above date. The level of approval is also listed above. Check the “Latest Version” or “Latest Approved 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.

The non-normative errata page for this specification is located at http://www.oasis-open.org/committees/WS-Calendar/.

Notices

Table of Contents

Tables

Index of Tables

Index of Examples

1 Introduction

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

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

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

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

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

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

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

1.1 Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [**RFC2119**].

1.2 Normative References

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

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

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

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

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

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

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

FreeBusy Read URL E York. *Freebusy read URL*, http://www.calconnect.org/pubdocs/CD0903%20Freebusy%20Read%20URL%20V1.0.pdf

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

Web-Linking M. Nottingham, *Web linking.* http://tools.ietf.org/html/draft-nottingham-http-link-header May 2010

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

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

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

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

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

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

1.3 Non-Normative References

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 January 2010.

NAESB Smart Grid Requirements *(dunno what reference I need here)*

REST T Fielding, *Architectural Styles and the Design of Network-based Software Architectures*, http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm.

1.4 Naming Conventions

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

For the names of elements and the names of attributes within XSD files, the names follow the CamelCase convention, with all names starting with a lower case letter, eg

```
<element name="componentType" type="WS-Calendar:ComponentType"/>
```

For the names of types within XSD files, the names follow the CamelCase convention with all names starting with an upper case letter, e.g.,

```
<complexType name="ComponentService">
```

For the names of intents, the names follow the CamelCase convention, with all names starting with a lower case letter, EXCEPT for cases where the intent is to represent an established acronym, in which case the entire name follows the usage of the established acronym.

An example of an intent which references an acronym is the "SOAP" intent.

1.5 Architectural References

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

2 Overview of WS-Calendar

A calendar communication without a real world effect[1] is of little interest. That real world effect is the result of a services execution context within a policy context. Practitioners can use WS-Calendar to add communication of schedule and interval to the execution context of a service. Use of WS-Calendar will align the performance expectations between execution contexts in different domains. The Technical Committee intends for other specifications and standards to incorporate WS-Calendar, bringing a common scheduling context to diverse interactions in different domains

2.1 Approach taken by the WS-Calendar Technical Committee

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

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

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

2.2 Overview of This Document

The specification consists of a standard schema and semantics for schedule and interval information. These semantic elements are defined and discussed in Section 3.

Often the most important service schedule communications involve series of related services over time, which WS-Calendar defines as a Series. Section 4 discusses the construction of series, and the association of service attributes to an entire series.

Within an iCalendar message, there is a larger document envelope containing transaction and synchronization information. This information is used for interactions between schedules, calendars, and calendar collections. The specification defines services for calendar inquiries, event scheduling, event updating, and event cancelation. RESTful service interactions for scheduling and interactions with calendars are described in sections 5-nn.

The standard also includes guidance for including geo-location within an event.

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

3 WS-Calendar Semantics

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

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

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

3.1 Scheduling Service Performance

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

As defined in the OASIS Reference Model for Service Oriented Architecture 1.0[2], service requests access the capability of a remote system.

> *The purpose of using a capability is to realize one or more real world effects. At its core, an interaction is "an act" as opposed to "an object" and the result of an interaction is an effect (or a set/series of effects). This effect may be the return of information or the change in the state of entities (known or unknown) that are involved in the interaction.*
>
> *We are careful to distinguish between public actions and private actions; private actions are inherently unknowable by other parties. On the other hand, public actions result in changes to the state that is shared between at least those involved in the current execution context and possibly shared by others. Real world effects are, then, couched in terms of changes to this shared state*

A request for remote service performance is a request for specific real world effects. Consider two service providers that offer the same service. One must start planning an hour or more in advance. The second may be able to achieve the service in five minutes. The service start time is the time when that service becomes available. If we do not distinguish these circumstances, then the customer would receive quite different quite different services with no distinctions in the service contract.

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

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

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

3.2 Core Semantics derived from [XCAL]

The iCalendar data format [[RFC5545] is a widely deployed interchange format for calendaring and scheduling data. The [XCAL] specification (in process) standardizes the XML representation of iCalendar

[2] See normative references in section 1.2

information. WS-Calendar relies on [XCAL] standards and data representation to develop its semantic components.

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

3.2.1 Time

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

```
Year:
      YYYY (eg 1997)
   Year and month:
      YYYY-MM (eg 1997-07)
   Complete date:
      YYYY-MM-DD (eg 1997-07-16)
   Complete date plus hours and minutes:
      YYYY-MM-DDThh:mmTZD (eg 1997-07-16T19:20+01:00)
   Complete date plus hours, minutes and seconds:
      YYYY-MM-DDThh:mm:ssTZD (eg 1997-07-16T19:20:30+01:00)
   Complete date plus hours, minutes, seconds and a decimal fraction of a
second
      YYYY-MM-DDThh:mm:ss.sTZD (eg 1997-07-16T19:20:30.45+01:00)
```

Normative information on ISO 8601 is referenced in section 1.2.

In WS-Calendar, unless otherwise noted, all times are un Greenwich Mean Time (UTC).

3.2.2 The iCalendar Components (VComponents)

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

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

3.2.3 Intervals

Time Segments, i.e., increments of continuous passage of time, are a critical component of service alignment using WS-Calendar. There are many overloaded uses of terms about time, and within a particular time segment, there may be many of them. Within this document, we use the term Time Segments to encompass all the terms in Table 1, below.

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

Table 1: Defining Time Segments for WS-Calendar

Time Segment	Definition
Duration	Well-known element from iCalendar and [XCAL], Duration is the length of an event scheduled using iCalendar or any of its derivatives. The [XCAL] duration is a data type using the string representation defined in the iCalendar duration. The Duration is the sole descriptive element of the VTODO object that is mandatory in the Interval.

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

3.2.4 Relationships between Intervals

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

```
<relationship>
    <uid>aaaaaaa1</uid>
    <reltype>PARENT</reltype>
</relationship>
```

WS-Calendar instead uses the reltype as an attribute of a relationship to support more expressive XSD annotation, like this:

```
<relationship reltype= "PARENT">
    <uid>aaaaaaa1</uid>
</relationship>
```

This format more easily supports the more expressive relationships used in Sequences.

WS-Calendar defines additional relationships to support temporal relationships between intervals. The relationships express the order of performance and to declare the spacing between those intervals. These relationships are referred to as the temporal relationships between components.

Table 2: Temporal Relationships in WS-Calendar

Temporal Relationship	Short Form	Definition
FINISHSTART	FS	As soon as the related Component finishes, this interval begins.
FINISHFINISH	FF	Used without gap when to components must finish at the same time. If there is a gap, it indicates that the referring component will finish execution a duration after the referred-to component.
STARTFINISH	SF	This component must Finish before the related component starts.

STARTSTART	SS	These Components must start at the same time
Gap		Attribute to indicate the separation, if any, between the state of the first Interval and the state of the second. Expressed as a duration.

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

Table 3: Elements of a Temporal Relationship

Relationship Element		Definition
reltype	String, Mandatory	Enumerated list from union of iCalendar and WS-Calendar Temporal Relationships.
guid	string Optional	Identifier of Component in Components collection. Mandatory if uri not present.
reference	[XPOINTER]	Reference to component external to components collection. Mandatory if uri not present.
gap	Duration *Optional*	Attribute to indicate the separation, if any, between the state of the first Interval and the state of the second. Expressed as a duration. Only used with Temporal Relationships

The relationship below indicates that this Interval is to start ten minutes following the finish of interval aaaaaaa1.

Example 1: Temporal Relationship

```
<relationship type="FINISHSTART">
  <uid>aaaaaaa1</uid>
  <gap>T00:10</gap>
</relationship>
```

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

Example 2: Temporal Relationship with and without Gap

```
<relationship type="STARTSTART">
  <uid>aaaaaaa1</uid>
  <gap>T00:00</gap>
</relationship>
```

Leaving out the optional Gap element, we have:

```
<relationship type="STARTSTART">
  <uid>aaaaaaa1</uid>
</relationship>
```

These two expressions of a Temporal Relationship above are equivalent.

Intervals with Temporal Relationships enable the message to express a Sequences as consecutive Intervals, as in a Partition, or they may express more complex temporal relations.

As the rules for parsing XML do not mandate preservation of order within a sub-set, we cannot assume that order is preserved when parsing a set of Components. For Sequences, mere order is not enough—each Interval must either refer to or be referred by at least one interval.

3.2.5 Alarms

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

3.3 Notification and Synchronization

An alarm notifies another party that something has happened. Some alarms, such as alarm clocks, are scheduled explicitly. Others arise as a surprise from another system. Actual alarm mechanisms and communications are outside the scope of this document. WS-Eventing, oBIX alarms, and CAP and EDXL alerts are just a few of the already defined mechanisms.

This section discusses how the iCalendar VALARM object is used in WS-Calendar. Alarms in a client server world are receiving a lot of attention in enterprise scheduling right now and some details may change before final publication.

A "VALARM" calendar component is a grouping of component properties that is a reminder or alarm for an event or a to-do. For example, it may be used to define a reminder for a pending event or an overdue to-do. The "VALARM" calendar component MUST include the "ACTION" and "TRIGGER" properties. . .The "ACTION" property is used within the "VALARM" calendar component to specify the type of action invoked when the alarm is triggered. The "VALARM" properties provide enough information for a specific action to be invoked3.

In WS-Calendar, an alarm is a VALARM object within a VTODO object, Its actions are [XPOINTER] references to the service or event that is triggered,

Valarm also supports recurring activities. A long-running VTODO service could be started alongside a recurring call-out to a 3rd service providing observation of the service's effects. For example, a Demand Response VTODO could be launched accompanied by a recurring 5 minute request to read the meter from another service.

3.4 Time Stamps

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

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

Table 4: Aspects of Time Stamps

Time Stamp Element	Definition (Normative)	Note (Non-Normative)
timeStamp	WS-Calendar:time A fully qualified date and time of event. Mandatory.	May include two objects as defined above.

3 From the [RFC5545] – see normative references

Time Stamp Element	Definition (Normative)	Note (Non-Normative)
precision	A Duration defining the accuracy of the TimeStamp value. Mandatory.	Identifies whether one hour interval is indeed one hour or plus or minus some number of milliseconds, seconds and minutes.
timeStampRealm	Of type Uri, shall identify the system where the TimeStamp value originated. The value of this element shall be set by: • The component at the realm border in a particular inter-domain interaction or, • By any component able to accurately set it within a system or sub-system. In the latter case, nothing prevents the component at the realm border to overwrite it without any notice. Optional.	A set of points originating from the same realm are reasonably synchronized. Within a realm, one can assume that time-stamped objects sorted by time are in the order of their occurrence. Between realms, this assumption is rebuttable. A system border is crossed in an interaction when the 2 communication partners are not synchronized based on the same time source. See the example below for more information.
leapSecondsKnown	Xs:bool If True, shall indicate that the TimeStamp value takes into account all leap seconds occurred. Otherwise False. Optional.	Indicates that the time source of the sending device support leap seconds adjustments.
clockFailure	xs:bool If True, shall indicate a failure on the time source preventing the TimeStamp value issuer from setting accurate timestamps. Otherwise False. Mandatory.	Indicates that the time source of the sending device is unreliable. The timestamp should be ignored.
clockNotSynchronized	xs:bool If True, shall indicate the time source of the TimeStamp value issuer is not synchronized correctly, putting in doubt the accuracy of the timestamp. Mandatory.	Indicates that the time source of the sending device is not synchronized with the external UTC time source.
timeSourceAccuracy	A Duration defining the accuracy of the time source used in the TimeStampRealm system. Optional.	Represents the time accuracy class of the time source of the sending device relative to the external UTC time source.

3.4.1 Time Stamp Realm Discussion

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

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

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

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

4 WS-Calendar Semantics

4.1 Services and Service Characteristics

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

In iCalendar, each component as an ATTACH element to carry unstructured information elaborating the event or alarm communication. Attachments in iCalendar can also be in the form of URIs pointing outside the iCalendar structure.WS-Calendar uses structured XML to communicate the substance of the request. The details of that xml artifact are domain-specific and are outside the scope of this document.

4.1.1 Attachments

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

Table 5: Elements of an WS-Calendar Attachment

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

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

Example 3: Use of an Attachment with inline XML artifact

```
<VTODO>
  <dtstamp></dtstamp>
  <uid>aaaaaaa1</uid>
  <description>first contract</description>
  <summary>defines contract to invoke Hello World Service</summary>
  <duration>T00:15</duration>
  <attach>
    <process name="pns:HelloWorld>
      <active>TRUE</active>
      <service name="wns:HelloWorldService" port="HelloWorldPort"/>
    </process>
  </attach>
```

```
</VTODO>
```

Example 4: Use of an Attachment with external reference

```
<VTODO>
  <dtstamp></dtstamp>
  <uid>aaaaaaa1</uid>
  <description>first contract</description>
  <summary>defines contract to described at reference</summary>
  <duration>T00:15</duration>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract1<reference>
  </attach>
</VTODO>
```

4.1.2 Specifying Timely Performance

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

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

Table 6: Performance Characteristics

Performance Characteristic	Definition	Discussion
StartBeforeTolerance	A Duration enumerating how far before the requested start time the requested service may commence.	Indicates if a service that begins at 1:57 is compliant with a request to start at 2:00
StartAfterTolerance	A Duration enumerating how far after the requested start time the requested service may commence.	Indicates if a service that begins at 2:01 is compliant with a request to start at 2:00
EndBeforeTolerance	A Duration enumerating how far before scheduled end time may end.	Indicates if a service that ends at 1:57 is compliant with a request to end at 2:00
EndAfterTolerance	A Duration enumerating how far after the scheduled end time the requested service may commence.	Indicates if a service that ends at 2:01 is compliant with a request to end at 2:00
DurationLongTolerance	A Duration indicating by how much the performance duration may exceed the duration specified in the Interval . It may be 0.	Used when run time is more important than start and stop time. DurationLongTolerance SHALL NOT be used when Start and End Tolerances are both specified.
DurationShortTolerance	A Duration indicating by how much the performance duration may fall short of duration specified in the Interval . It may be 0.	Used when run time is more important than start and stop time. DurationShortTolerance SHALL NOT be used when Start and End Tolerances are both specified.

Performance Characteristic	Definition	Discussion
Granularity	A Duration enumerating the smallest unit of time measured or tracked	Whatever the time tolerance above, there is some minimum time that is considered insignificant. A Granularity of 1 second defines the tracking and reporting requirements for a service.

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

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

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

Example 5: Performance Component

```
<performance>
    <startbefore>T00:10</startbefore>
    <startafter>T00:00</startafter>
    <durationlong>T00:00</durationlong>
    <durationshort>T00:00</durationshort>
</performance>
```

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

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

4.1.3 Combining Service and Performance

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

Example 6: Use of an Attachment with inline XML artifact and optional specified Performance

```
<VTODO>
  <dtstamp></dtstamp>
  <uid>aaaaaaa1</uid>
  <description>first contract</description>
  <summary> defines contract to invoke Hello World Service as early as 10
minutes before scheduled time, and no later than scheduled time</summary>
  <duration>T00:15</duration>
  <attach>
    <process name="pns:HelloWorld>
      <active>TRUE</active>
      <service name="wns:HelloWorldService" port="HelloWorldPort"/>
    </process>
    <performance>
      <startbefore>T00:10</startbefore>
      <startafter>T00:00</startafter>
      <durationlong>T00:00</durationlong>
      <durationshort>T00:00</durationshort>
    </performance>
  </attach>
</VTODO>
```

Example 7: Use of an Attachment with external reference and optional specified performance

```
<VTODO>
```

```
  <dtstamp></dtstamp>
  <uid>aaaaaaa1</uid>
  <description>first contract</description>
  <summary>defines first behavior to perform in contract with a precisions
required of 1 second</summary>
  <duration>T00:15</duration>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract1<reference>
    <performance>
      <startbefore>T00:10</startbefore>
      <startafter>T00:00</startafter>
      <durationlong>T00:00</durationlong>
      <durationshort>T00:00</durationshort>
    </performance>
  </attach>
</VTODO>
```

5 WS-Calendar Semantics

Calendar Gluons and Inheritance

5.1.1 Calendar Gluons

WS-Calendar introduces a new iCalendar component, the Calendar Gluon. An Calendar Gluon is essentially a placeholder vComponent (see Appendix *Overview of WS-Calendar, its Antecedents and its Use*) used to assign attributes to an entire Sequence. Calendar Gluons use the RelatedComponent attribute to apply service information to Sequences and Partitions. The use of Calendar Gluons is described in *Section* ***Error! Reference source not found.***: ***Error! Reference source not found.***.

Table 7: Calendar Gluon elements in WS-Calendar

Calendar Gluon Element	Use	Discussion
dtStamp	[XCAL]:dtstamp *Mandatory*	Time and date that Calendar Gluon object was created
Uid	*Mandatory*	Used to enable unambiguous referencing of each VTODO object
summary	Text' *Optional*	Text describing the Calendar Gluon
Related	WsCalendar:Relationship *Mandatory*	An Calendar Gluon must have a relationship with at least one other component. The only relationship defined for the Calendar Gluon is the IsParent.
dtStart	[XCAL]:Time. Start time for the related interval of the sequence. *Optional*	An Calendar Gluon may either have a dtStart or a dtEnd, but may not have both.
dtEnd	[XCAL]:Time. Scheduled completion time for the related interval of the sequence. *Optional*	An Calendar Gluon may either have a dtStart or a dtEnd, but may not have both.
duration	[XCAL]:Duration *Optional*	If specified, a duration is inherited by all intervals in the referred-to sequence,
Attach	WSCalendar:Attachment Mandatory Multipleoccurs	Contains WS-Calendar:attachment attribute defining service and performance. Can be inherited by all intervals in sequence.

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

Inheritance is discussed in greater detail in Chapter 4.

5.1.2 Calendar Gluons and Sequences

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

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

Example 8: Sequence with Performance defined in the Calendar Gluon

```
<components>
<Calendar Gluon>
  <dtstamp></dtstamp>
  <uid>aaaaaaa0</uid>
  <description>Calendar Gluon with sequence</description>
  <summary>creates common performance expectations (+/- 1 second) for
     entire sequence. Also sets common duration for all members of the
     sequence.
  </summary>
  <duration>T00:15</duration>
  <attach>
    <performance>
      <endbefore>T00:00</endbefore>
      <endafter>T00:00</endafter>
      <durationlong>T00:00</durationlong>
      <durationshort>T00:00</durationshort>
    </performance>
  </attach>
  <related-to>
    <relationship type="PARENT">
      <uid>aaaaaaa1</uid>
    </relationship>
  </related-to>
</Calendar Gluon>
<vtodo>
  <dtstamp></dtstamp>
  <uid>aaaaaaa1</uid>
  <description>first contract</description>
  <summary>inherits performance expectations & duration</summary>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract1<reference>
  </attach>
</vtodo>
<vtodo>
  <dtstamp></dtstamp>
  <uid>aaaaaaa2</uid>
  <description>second interval</description>
  <summary>inherits performance expectations & duration</summary>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract2<reference>
  </attach>
  <related-to>
    <relationship type="FINISHSTART">
      <uid>aaaaaaa1</uid>
    </relationship>
    <relationship>
  </related-to>
</vtodo>
```

```
<vtodo>
  <dtstamp></dtstamp>
  <uid>aaaaaaa3</uid>
  <description>third interval</description>
  <summary>inherits performance expectations, overrides duration</summary>
  <duration>T00.30</duration>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract3<reference>
  </attach>
  <related-to>
    <relationship type="FINISHSTART">
      <uid>aaaaaaa2</uid>
      <gap>T00:10</gap>
    </relationship>
  </related-to>
</vtodo>
<components>
```

This sequence is functionally identical to the one before. Note that the performance expectations, identical for each interval, have moved into the Calendar Gluon.

The Calendar Gluon happens to related to the first Interval in the sequence; there are specific use cases (discussed below) which require it to be linked to other Intervals. As a Sequence creates single temporal relationship, assigning a start time (dtstart) to any Interval allows the starting time to be computed for any of them.

5.1.3 Optimizing the Sequence for a Partition

Partitions are sequences with consecutive Intervals. Partition communication can be further optimized by bringing the relationship into the Calendar Gluon. Notice that while the type of the relationship is defined in the Calendar Gluon, the guid for each interval must still be expressed within the interval.

Example 9: Partition with Duration and Performance defined in the Calendar Gluon

```
<components>
<Calendar Gluon>
  <dtstamp></dtstamp>
  <uid>aaaaaaa0</uid>
  <description>Calendar Gluon with sequence</description>
  <summary>creates common performance expectations (+/- 1 second) for
     entire sequence. Also sets common duration for all members of the
     sequence.
  </summary>
  <duration>T00:50</duration>
  <attach>
    <performance>
      <startbefore>T00:00</endbefore>
      <startafter>T00:05</endafter>
      <durationlong>T00:00</durationlong>
      <durationshort>T00:05</durationshort>
    </performance>
  </attach>
  <related-to>
    <relationship type="PARENT">
      <uid>aaaaaaa1</uid>
    </relationship>
    <relationship type="FINISHSTART">
      <gap>T00:10</gap>
    </relationship>
  </related-to>
</Calendar Gluon>
<vtodo>
  <dtstamp></dtstamp>
  <uid>aaaaaaa1</uid>
```

```
  <description>first contract</description>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract1<reference>
  </attach>
</vtodo>
<vtodo>
  <dtstamp></dtstamp>
  <uid>aaaaaaa2</uid>
  <description>second interval</description>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract2<reference>
  </attach>
  <related-to>
    <relationship><uid>aaaaaaa1</uid></relationship>
  </related-to>
</vtodo>
<vtodo>
  <dtstamp></dtstamp>
  <uid>aaaaaaa3</uid>
  <description>third interval</description>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract3<reference>
  </attach>
  <related-to>
    <relationship><uid>aaaaaaa2</uid></relationship>
  </related-to>
</vtodo>
<components>
```

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

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

*Example 10: Partition in **Error! Reference source not found.** without annotations*

```
<components>
<Calendar Gluon>
  <dtstamp></dtstamp>
  <uid>aaaaaaa0</uid>
  <duration>T00:50</duration>
  <attach>
    <performance>
      <startbefore>T00:00</endbefore>
      <startafter>T00:05</endafter>
      <durationlong>T00:00</durationlong>
      <durationshort>T00:05</durationshort>
    </performance>
  </attach>
  <related-to>
    <relationship type="PARENT">
      <uid>aaaaaaa1</uid>
    </relationship>
    <relationship type="FINISHSTART">
      <gap>T00:10</gap>
    </relationship>
  </related-to>
</Calendar Gluon>
<vtodo>
  <uid>aaaaaaa1</uid>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract1<reference>
  </attach>
</vtodo>
```

```
<vtodo>
  <uid>aaaaaaa2</uid>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract2<reference>
  </attach>
  <related-to>
    <relationship><uid>aaaaaaa1</uid></relationship>
  </related-to>
</vtodo>
<vtodo>
  <uid>aaaaaaa3</uid>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract3<reference>
  </attach>
  <related-to>
    <relationship><uid>aaaaaaa2</uid></relationship>
  </related-to>
</vtodo>
<components>
```

Notice that the dtstamp for all Intervals in this Partition is inherited from the Calendar Gluon.

5.1.4 Scheduling a Sequence

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

Example 11: A Scheduled Sequence

```
<components>
<vtodo>
  <dtstamp></dtstamp>
  <uid>aaaaaaa1</uid>
  <description>first contract</description>
  <dtstart>2010-09-11T13:00</dtstart>
  <duration>T00:15</duration>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract1<reference>
  </attach>
</vtodo>
<vtodo>
  <dtstamp></dtstamp>
  <uid>aaaaaaa2</uid>
  <description>second interval</description>
  <duration>T00:15</duration>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract2<reference>
  </attach>
  <related-to>
    <relationship type="FINISHSTART">
      <uid>aaaaaaa1</uid>
    </relationship>
    <relationship>
  </related-to>
</vtodo>
<components>
```

A sequence can also be scheduled in the Calendar Gluon.

Example 12: A Scheduled Sequence showing Temporal Relationship Inheritance

```
<components>
<Calendar Gluon>
  <dtstamp></dtstamp>
  <uid>aaaaaaa0</uid>
```

```
  <dtstart>2010-09-11 T00:15</dtstart>
  <related-to>
    <relationship type="PARENT">
      <uid>aaaaaaa1</uid>
    </relationship>
    <relationship type="FINISHSTART">
      <gap>T00:10</gap>
    </relationship>
  </related-to>
</Calendar Gluon>
<vtodo>
  <dtstamp></dtstamp>
  <uid>aaaaaaa1</uid>
  <description>first contract</description>
  <duration>T00:15</duration>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract1<reference>
  </attach>
</vtodo>
<vtodo>
  <dtstamp></dtstamp>
  <uid>aaaaaaa2</uid>
  <description>second interval</description>
  <duration>T00:15</duration>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract2<reference>
  </attach>
  <related-to>
    <relationship type="FINISHSTART">
      <uid>aaaaaaa1</uid>
    </relationship>
    <relationship>
  </related-to>
</vtodo>
<components>
```

5.1.5 Mixed Inheritance of Start Time

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

To illustrate this, here is the classroom scheduling Partition from **Error! Reference source not found.**, updated to include each day's school opening.

Example 13: Partition with Duration and Performance defined in the Calendar Gluon

http://scheduled.ws-calendar-service.com/classSchedule

```
<components>
<Calendar Gluon>
  <dtstamp></dtstamp>
  <uid>aaaaaaa0</uid>
  <dtstart>T19:00</dtstart>
  <description>Classroom Schedule</description>
  <duration>T00:50</duration>
  <related-to>
    <relationship type="PARENT">
      <uid>aaaaaaa1</uid>
    </relationship>
    <relationship type="FINISHSTART">
      <gap>T00:10</gap>
    </relationship>
  </related-to>
```

```
</Calendar Gluon>
<vtodo>
  <dtstamp></dtstamp>
  <uid>aaaaaaa1</uid>
  <description>first interval</description>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract1<reference>
  </attach>
</vtodo>
<vtodo>
  <dtstamp></dtstamp>
  <uid>aaaaaaa2</uid>
  <description>second interval</description>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract2<reference>
  </attach>
  <related-to>
    <relationship><uid>aaaaaaa1</uid></relationship>
  </related-to>
</vtodo>
<vtodo>
  <dtstamp></dtstamp>
  <uid>aaaaaaa3</uid>
  <description>third interval</description>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract3<reference>
  </attach>
  <related-to>
    <relationship><uid>aaaaaaa2</uid></relationship>
  </related-to>
</vtodo>
<components>
```

and the invoking Calendar Gluon for a given day:

```
<components>
<Calendar Gluon>
  <dtstamp></dtstamp>
  <uid>aaaaaaac</uid>
  <dtstart>2010-09-11</dtstart>
  <related-to>
    <relationship type="PARENT">
      <reference>http://scheduled.ws-calendar-service.com/classSchedule
      </reference>
    </relationship>
  </related-to>
</Calendar Gluon>
<components>
```

In this case, the Sequence is offered at 13:00. The Sequence is not yet scheduled because a schedule requires a full start date and time. The Sequence has an external reference. The Calendar Gluon schedules a particular performance of this sequence on 2010-09-11. The date from the invocation and the time from the offering are combined to produce 2010-09-11T13:00 and the result is a Scheduled Sequence.

5.1.6 Other Scheduling Scenarios

Sometime the invoker of a service is interested only in single Interval of the Sequence, but the entire Sequence is required. In this case, it is valuable to invoke the Sequence by a particular interval.

Example 14: Standard Sequence with Ramp-Up and Ramp Down

http://scheduled.ws-calendar-service.com/anotherSchedule

```
<components>
```

```
<Calendar Gluon>
  <dtstamp></dtstamp>
  <uid>aaaaaaa0</uid>
  <dtstart>T19:00</dtstart>
  <description>Note reference to second interval</description>
  <duration>T00:50</duration>
  <related-to>
    <relationship type="PARENT">
      <uid>aaaaaaa2</uid>
    </relationship>
    <relationship type="FINISHSTART">
      <gap>T00:10</gap>
    </relationship>
  </related-to>
</Calendar Gluon>
<vtodo>
  <dtstamp></dtstamp>
  <uid>aaaaaaa1</uid>
  <description>first interval</description>
  <summary>set up process required before second interval can be performed.
      setup takes a fixed time
  </summary>
  <duration>T00:50</duration>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract1<reference>
  </attach>
</vtodo>
<vtodo>
  <dtstamp></dtstamp>
  <uid>aaaaaaa2</uid>
  <description>second interval</description>
  <summary>This is the one that the invoker is interested in. The invoker
      may invoke this event for whatever period desired. Note that the
      external Calendar Gluon is parent to this Interval
  </summary>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract2<reference>
  </attach>
  <related-to>
    <relationship><uid>aaaaaaa1</uid></relationship>
  </related-to>
</vtodo>
<vtodo>
  <dtstamp></dtstamp>
  <uid>aaaaaaa3</uid>
  <description>third interval</description>
  <summary>however long the second interval takes, this interval must run
     take 15 minutes for completion.
  </summary>
  <duration>T00:15</duration>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract3<reference>
  </attach>
  <related-to>
    <relationship><uid>aaaaaaa2</uid></relationship>
  </related-to>
</vtodo>
<component>
```

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

```
<components>
<Calendar Gluon>
```

```
  <dtstamp></dtstamp>
  <uid>aaaaaaac</uid>
  <dtstart>2010-09-11T19:00</dtstart>
  <duration>T01:15</duration>
  <description>Classroom Schedule</description>
  <duration>T00:50</duration>
  <related-to>
    <relationship type="PARENT">
      <reference>http://scheduled.ws-calendar-service.com/classSchedule
      <uid>aaaaaaa1</uid>
    </relationship>
  </related-to>
</Calendar Gluon>
```

In this case, the specific interval is scheduled and a run time of 75 minutes is specified.

6 Calendar Service Interactions: Overview

This OASIS Committee has worked closely with the CalConnect TC-XML committee, which publishes its work through the IETF[4]. CalConnect is defining the core scheduling service interactions, i.e., scheduling an event, determining availability, etc., and publishing them as Cal-WS.

6.1 Glossary

3.1 Hrefs 8
3.2 Calendar Object Resource 8
3.3 Calendar Collection 8
3.4 Scheduling Calendar Collection ......

6.2 Issues not addressed by this specification

2.1 Access Control ... 7
2.2 Creating Collections 7
2.3 Provisioning 7
2.4 Discovery 7
2.5 Retrieving collections 7

[4] http://datatracker.ietf.org/wg/calsify/charter/

7 Service Interactions: Protocol Overview

8 Service Capabilities

Different Calendars and schedule systems have different capabilities. The more sophisticated system may have to simplify interactions to interact with the less capable system.

9 Creating Calendar Resources

10 Retrieving Calendar Resources

11 Updating Calendar Resources

12 Deletion of Calendar Resources

13 Querying Calendar Resources

14 Conformance

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

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

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

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

All OASIS specifications require conformance

A. Acknowledgements

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

Participants:

Brad Benson, Trane
Edward Cazalet, Individual
Toby Considine, University of North Carolina at Chapel Hill
William Cox, Individual
Craig Gemmill, Tridium, Inc.
Girish Ghatikar, Lawrence Berkeley National Laboratory
Gale Horst, Electric Power Research Institute (EPRI)
Gershon Janssen, Individual
Ed Koch, Akuacom Inc.
Benoit Lepeuple, LonMark International*
Carl Mattocks, CheckMi*
Robert Old, Siemens AG
Alexander Papaspyrou, Technische Universitat Dortmund
Jeremy Roberts, LonMark International*
David Thewlis, CalConnect

The Calendaring and Scheduling Consortium (CalConnect) TC-XML committee worked closely with WS-Calendar Technical Committee, bridging to developing IETF standards and contributing the Services definitions that make up Section 6, Calendar Service Interactions. The Technical Committee gratefully acknowledges their assistance and cooperation as well.

B. Understanding iCalendar, it history, and its use

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

Non normative stuff coming from CalConnect

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

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

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

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

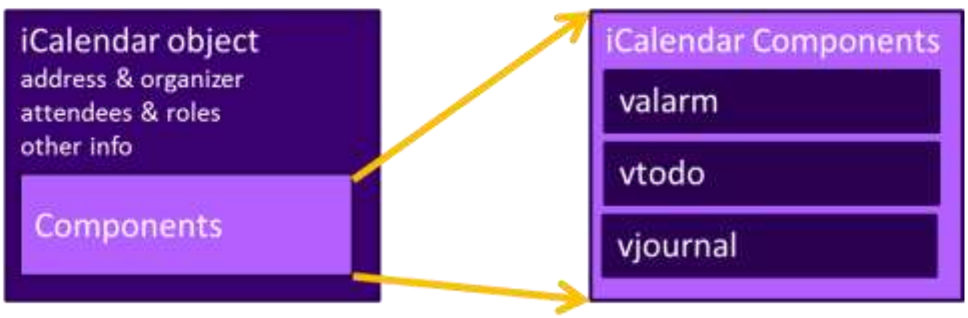

Figure 1: iCalendar overview

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

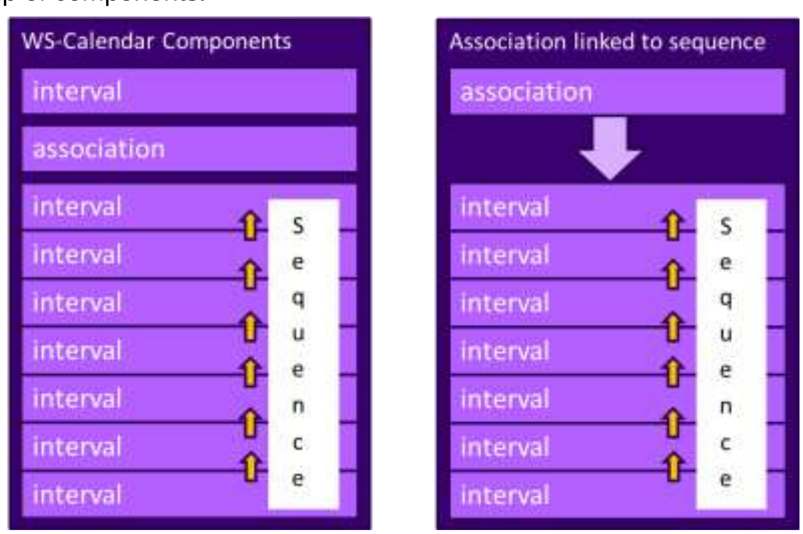

Figure 2: WS-Calendar and EMIX

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

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

C.1 Scheduling Sequences

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

C.1.1 Academic Scheduling example

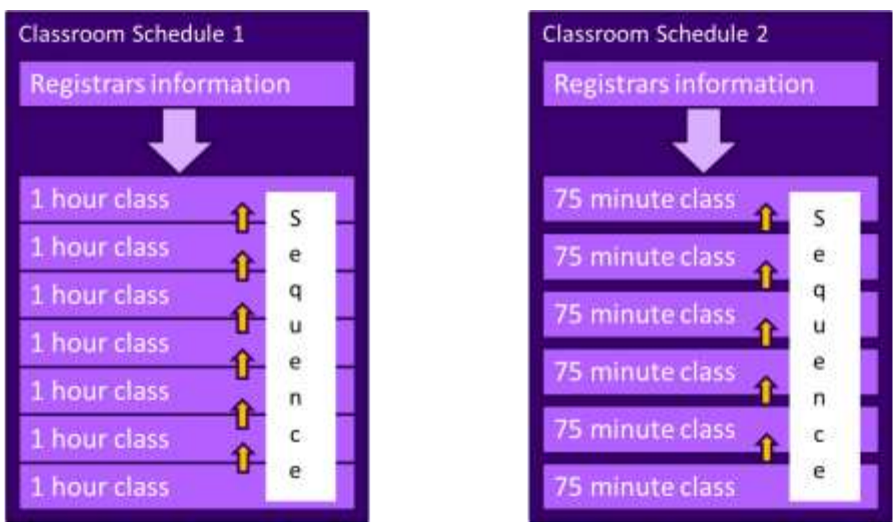

Figure 3: Classroom Scheduling Example

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

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

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

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

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

C.1.2 Market Performance schedule

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

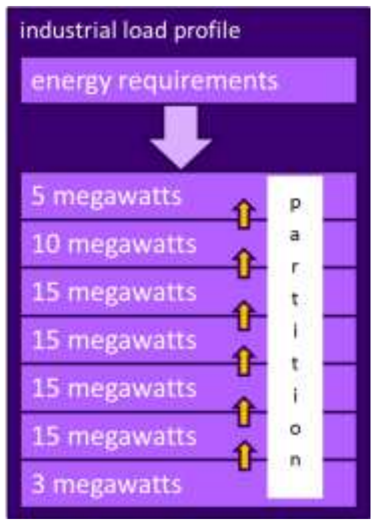

Figure 4: Daily Load Profile for Market Operations Example

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

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