WS-Calendar Version 1.0

Working Draft 09

15 August 2010

Specification URIs:
This Version:
http://docs.oasis-open.org/WS-Calendar/v1.0/wd09/WS-Calendar-1.0-spec-wd-09.pdf
http://docs.oasis-open.org/WS-Calendar/v1.0/wd09/WS-Calendar-1.0-spec-wd-09.html
http://docs.oasis-open.org/WS-Calendar/v1.0/wd09/WS-Calendar-1.0-spec-wd-09.doc
Previous Version:
http://docs.oasis-open.org/WS-Calendar/v1.0/wd08/WS-Calendar-1.0-spec-wd-08.pdf
http://docs.oasis-open.org/WS-Calendar/v1.0/wd08/WS-Calendar-1.0-spec-wd-08.html
http://docs.oasis-open.org/WS-Calendar/v1.0/wd08/WS-Calendar-1.0-spec-wd-08.doc
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 RFC 5545, ICalendar
- IETF RFC 5546, ICalendar Transport
- IETF RFC 2447, 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 to Tables

Table 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 is built on work done and ongoing in The Calendaring and Scheduling Consortium (CalConnect), which works to increase interoperation between calendaring systems.

A calendar communication without a real world effect is of little interest. That real world effect is the result of a services execution context within a policy context[1]. 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.

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

[1] See *“Reference Model for Service Oriented Architecture 1.0”* for definitions of all terms used herein to describe service interactions.

1.1 Terminology

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

1.2 Normative References

RFC2119 S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*, http://www.ietf.org/rfc/rfc2119.txt, IETF RFC 2119, March 1997.

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

RFC5545 B. Desruisseaux *Internet Calendaring and Scheduling Core Object Specification (iCalendar)*, http://www.ietf.org/rfc/rfc5545.txt, IETF RFC 5545, September 2009.

RFC5546 C. Daboo *iCalendar Transport-Independent Interoperability Protocol (iTIP)*, http://www.ietf.org/rfc/rfc5546.txt, IETF RFC 5546, January 1999.

RFC2447 F. Dawson, S. Mansour, S. Silverberg, *iCalendar Message-Based Interoperability Protocol (iMIP)*, http://www.ietf.org/rfc/rfc2247.txt, IETF RFC 2447, December 2009.

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.

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.

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.

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)*

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[2] 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 Committee based its work upon the iCalendar specification as updated in 2009 (IETF RFC 5545) and its the XML serialization xCal, currently (2010-07) on a standards track in the IETF. Both updates were to IETF specifications were developed by members of the Calendaring and Scheduling Consortium (CalConnect.org). 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

Based on these requirements, the technical committee developed the semantic elements in sections three and four.

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, which CalConnect refers to as CalWS, provide the basic interactions for querying, creating, updating, and deleting calendar events that are common to all calendars and schedules. CalConnect donated CalWS to WS-Calendar to make up the service interactions in section 5.

2.2 Specification Deliverables

The specification consists of a standard schema and semantics for schedule and interval information. The specification also includes standard service calls for calendar inquiries, event scheduling, event updating, and event cancelation. Finally, the specification includes rules for delivering a sequence of operations, i.e., a representation of several services that are scheduled as a single event.

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

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

3 WS-Calendar Definitions

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. In effect, the Elements are uded to build a precise vocabulary of time, duration, sequencing, and schedule.

The lexicon of Elements is also used to decorate and elaborate the simpler specification of xCal to make explicit the performance expectations within a scheduled event. xCal to standardize data and interval outside of scheduling interactions.

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.

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[3], 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 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

[3] 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.

3.2.2 The iCalendar Components (VObjects)

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 RFC 5545.

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 VObjects 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 a meeting scheduled using iCalendar or any of its derivatives. The xCal duration is a data type using the string representation defined in the iCalendar duration. The Duration is the sole descriptive element of the VTODO object that is mandatory in the Interval
Interval	The Interval is a single duration supported by the full information set of the VTODO object as defined in iCalendar (RFC 5545) and refined in xCal. A WS-Calendar interval must include a Duration.

Time Segment	Definition
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 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.

3.2.5 Related Components

WS-Calendar introduces a new iCalendar component, the RelatedComponent. A RelatedComponent is essentially a VObject with no schedule or interval elements. WS-Calendar uses RelatedComponents to apply service information to Sequences and Partitions. The use of Related Components is described in *Section 4: Intervals, Partitions, Sequences, Processes, and Process Synchronization*.

Table 2: RelatedComponent elements in WS-Calendar

Elements	Use	Discussion
Dtstamp	Mandatory	
Uid	Mandatory	Used to enable unambiguous referencing of each VTODO object
Class	Optional	
Summary	Optional	Text describing the Association
Attach	Mandatory, Multipleoccurs	Contains XML Artifact defining performance or xPointer to artifact defining performance. If repeated, can refer to multiple artifacts
Related	Mandatory	A RelatedComponent must have a relationship with at least one other component. The only relationship defined for the RelatedComponent is the IsParent.

3.3 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 eachof 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.

3.3.1 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 3: Elements of a WS-Calendar Attachments

Elements	Use	Discussion
Artifact	Optional. Any in-line XML. 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
Reference	Optional. XPOINTER. Must have at least one of Artifact or Reference	Points to external XML, or XML located elsewhere in document
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 1: 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 2: 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>
```

3.3.2 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 RFC 5545.

Table 4: 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.
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 3: 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.

3.3.3 Combining Service and Performance

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

Example 4: 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 5: 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>
```

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.

Different systems may track time and therefore record events with different levels of Tolerance. It is 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 5: Aspects of Time Stamps

WS-Calendar Time Stamp Element	Definition (Normative)	Note (Non-Normative)
TimeStamp	WS-Calendar:time A fully qualified date and time of event	May include two objects as defined above.
Precision	A Duration defining the accuracy of the time stamp.	Identifies whether one hour interval is indeed one hour or plus or minus some number of seconds and minutes
TimeStampRealm	The realm identifies where the time stamp originated Is this a GUID or some other sort of data type?	The term Realm originates in ISO 61850. A realm is a set of points which 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.
LeapSecondsKnown	Xs:bool	Need a definition here
ClockFailure	xs:bool	Indicates that the time source of the sending device is unreliable
ClockNotSynchronized	xs:bool	indicates that the time source of the sending device is not synchronized with the external UTC time
Accuracy	A Duration defining the accuracy of the clock used in the TimeStampRealm.	represents the time accuracy class of the time source of the sending device relative to the external UTC time.
Attachment	As defined in section **Error! Reference source not found.**	Contains either local description of service or reference to xml document describing service

4 Intervals, Partitions, Sequences, Processes, and Process Synchronization

WS-Calendar derives objects for communicating intervals and for synchronizing time from the corresponding iCalendar objects. Within an iCalendar message, there is a larger document envelope containing transaction and synchronization information. The use of those fields is discussed below in section 5 under Calendar Service Interactions.

In iCalendar (and therefore xCalendar), one of the top-level objects is the Components section which can contain one or many iCalendar components, the so-called VObjects. Traditional calendar sharing has tended to use only one or two components, say a single meeting (VEVENT) or perhaps a task (VTODO) and a request to warn the recipient of the impending due date in advance (VALARM).

Within WS-Calendar, these components can be strung together to create packages of service interactions and market operations. As services are advertised, they may not yet have specific performance time scheduled. For this reason, only the Duration is required.

A Start time plus a duration fully implies an end time; it is not necessary to specify a start, duration, and end. Specifying all three could allow to a message that is internally inconsistent. Allowing options leads to complexity. As the duration is the required element, one of the times is redundant. WS-Calendar specifies that only the Start Time and Duration are considered. While an end time is a legal component of a VTODO object, WS-Calendar services ignore it.

4.1 Use of VTODO elements

The simplest segment of time is a single interval. Intervals are derived from a single VTODO object. For ease of reference, the VTODO object is described below. In all cases, implementers SHALL refer to RFC 5545 and the xCal specifications for the normative description and definitions.

4.1.1 Use of VTODO elements

All elements of the VTODO component are legal in WS-Calendar, certain elements are more critical when invoking services. These elements and their definitions within WS-Calendar are listed in *Table 6: VTODO elements in* WS-Calendar. Elements marked mandatory SHALL be in every use of VTODO in WS-Calendar.

Table 6: VTODO elements in WS-Calendar

Elements	Use	Discussion
Dtstamp	Mandatory	
Uid	Mandatory	Used to enable unambiguous referencing of each VTODO object
Class	Optional	
Duration	Mandatory	
dtStart	Optional	Scheduled start date and time for interval
dtEnd	Ignored	Legal only when Duration is not specified. As WS-Calendar required Duration, dtEnd is ignored.
Attach	Mandatory, Multipleoccurs	Contains XML Artifact defining performance or xPointer to artifact defining performance. If repeated, can refer to multiple artifacts

Related	Optional compound element	Defines relations to other components. May be mandatory in derived specifications, esp. if component order is important.

4.1.2 Relationships between VTODO Objects

Many service communications involve more than one time segment. These segments may be consecutive, as in an Interval, or they may have a more complex temporal relation, as in Sequences. The rules for parsing XML do not mandate preservation of order within a sub-set. This means that we cannot assume that order is preserved when parsing a set of iCalendar Components. For Sequences, mere order is not enough—this leads to the relationships.

In iCalendar, each Component (a VObject in the Components Section) may have an array of relationships to the other Components. In WS-Calendar, each relationship may also have an optional Gap expressed as an iCalendar duration.

Example 6: Vobject Relationship

```
<relationship>
    <uid>aaaaaaa1</uid><reltype>FS</reltype><gap>T00:10</gap>
</relationship>
```

The Gap refines the relationship, in this cases, adding 10 minutes to the FS relationship. In the absence of a Gap, the Gap is assumed to be 0.

Table 7: Use of Inter-component Relationships in WS-Calendar

Relationship	Memnonic	Definition
FS	Finish-Start	As soon as the related Component finishes, this interval begins. This form is normally used in Intervals, i.e., consecutive Components If there is a gap, than it indicates the period between the finish of the referenced Component and the start of the referring Component.
FF	Finish-Finish	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.
SF	Start-Finish	This component must Finish before the related component starts. This relationship would be used by the preceding Component to refer to the next Component in a sequential Interval.
SS	Start-Start	These Components must start at the same time. If there is a gap, it refers to how long the referring Component has to start after the Component it references.

In an Interval, each component would have a FS relationship to the prior Component with no gap. In a Sequence, the relationships can be more complex.

4.2 Intervals and Sequences

An Interval specifies a single segment of time specified using a VTODO object. Sequences consist of one or more intervals. A Partition is a special case of a Sequence in which the Durations are identical and Intervals occur consecutively with no time in between.

Each VTODO in a Sequence may have a relationship with the other objects. These relationships detmine the temporal relationship between the Intervals. In Partitions, these relationships are limited to ordering the Intervals.

XML does not specify that sequence is maintained during XML processing. For this reason, even in the simple case of a Partition, each VTODO has a relationship its precedent and succedent. While we have included examples below, implementers should refer to RFC 5545 and the xCal specifications for the normative descriptions and definitions.

4.2.1 Intervals: the Basic Time Segment

An interval specifies how long an activity lasts. An unscheduled Interval is not linked to a specific date and time.

Table 8: Interval Data Elements

Elements	Use	Discussion
Dtstamp	Mandatory	
Uid	Mandatory	Used to enable unambiguous referencing of each VTODO object
Class	Optional	
Duration	Mandatory	
Performance	Optional	Defines performance characteristics of Interval. Ignored by traditional iCalendar processors.
Attach	Optional, Multipleoccurs	Contains XML Artifact defining service as specified in the Attachment section. Required unless a relationship is defined to a RelatedComponent that defines the service.
Related	Optional compound element	Defines relations to other components. Used to define temporal relationships. Can be used as external reference to service definition.

The example below shows the components section of a WS-Calendar event containing two consecutive 15 minute time segments. They are listed in order. As XML parsing is not guaranteed to result in the same order, each has a UID and a relationship. defining the order that each will occur. No start date is specified in these time segments. As they are linked to each other, they describe a service of 30 minutes total made up of two consecutive segments.

Example 7: An Interval

```
<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>
      <endbefore>T00:00</endbefore>
      <endafter>T00:00</endafter>
      <durationlong>T00:00</durationlong>
      <durationshort>T00:00</durationshort>
    </performance>
  </attach>
</VTODO>
```

Note that no start time is specified, and no relationship. Relationships are mandatory when an interval is incorporated into a Sequence.

4.2.2 Sequences: Putting things together

Sequences are collections of related Intervals. The relationships define the time relationships between the Intervals. Sequences become Scheduled Sequences when the first Interval is assigned a starting time.

Example 8: Simple sequence with three intervals

```
<components>
<VTODO>
  <dtstamp></dtstamp>
  <uid>aaaaaaa1</uid>
  <description>first contract</description>
  <priority>high</priority>
  <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>
      <endbefore>T00:00</endbefore>
      <endafter>T00:00</endafter>
      <durationlong>T00:00</durationlong>
      <durationshort>T00:00</durationshort>
    </performance>
  </attach>
  <related-to>
    <relationship>
      <uid>aaaaaaa2</uid><reltype>SF</reltype>
    </relationship>
  </related-to>
</VTODO>
<VTODO>
  <dtstamp></dtstamp>
  <uid>aaaaaaa2</uid>
  <description>second interval</description>
  <priority>high</priority>
  <summary>defines second behavior to perform in contract with a precision
required of 1 second</summary>
  <duration>T00:15</duration>
  <attach>
    <reference>http://scheduled.ws-calendar-service.com/contract2<reference>
    <performance>
      <endbefore>T00:00</endbefore>
      <endafter>T00:00</endafter>
      <durationlong>T00:00</durationlong>
      <durationshort>T00:00</durationshort>
    </performance>
  </attach>
  <related-to>
    <relationship>
      <uid>aaaaaaa1</uid><reltype>FS</reltype>
      <uid>aaaaaaa3</uid><reltype>SF+10m</reltype>
    </relationship>
  </related-to>
</VTODO>
<VTODO>
  <dtstamp></dtstamp>
  <uid>aaaaaaa3</uid>
  <description>second interval</description>
  <priority>high</priority>
  <summary>defines second behavior to perform in contract with a precision
required of 1 second</summary>
  <duration>T00.30</duration>
```

```
    <attach>
      <reference>http://scheduled.ws-calendar-service.com/contract3<reference>
      <performance>
        <endbefore>T00:00</endbefore>
        <endafter>T00:00</endafter>
        <durationlong>T00:00</durationlong>
        <durationshort>T00:00</durationshort>
      </performance>
    </attach>
    <related-to>
      <relationship>
        <uid>aaaaaaa2</uid><reltype>FS+T00.10</reltype>
      </relationship>
    </related-to>
  </VTODO>
  <components>
```

The first interval of 15 minutes is followed immediately by the second interval of 15 minutes. There is a 10 minute interval between the completion of the second interval and the beginning of the third.

In the example above, each Interval has its own performance characteristics.

4.2.3 Related Components and Sequences

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

This performance component defines that service may start up to 10 minutes early, may not start late, and an accuracty of two seconds is expected in all timing and reporting.

Example 9: Scheduled Sequence with shared performance definition

```
<components>
<RelatedComponent>
  <dtstamp></dtstamp>
  <uid>aaaaaaa0</uid>
  <summary>Defines performance requirements for entire sequence</summary>
  <attach>
    <performance>
      <startbefore>T00:10</startbefore>
      <startafter>T00:00</startafter>
      <durationlong>T00:00</durationlong>
      <durationshort>T00:00</durationshort>
      <granularity>T00:00:02</dgranularity>
    </performance>
  </attach>
    <relationship>
      <uid>aaaaaaa1</uid><reltype>Parent</reltype>
    </relationship>
</RelatedComponent>
<VTODO>
  <dtstamp></dtstamp>
  <uid>aaaaaaa1</uid>
  <description>first contract</description>
  <priority>high</priority>
  <summary>defines first behavior to perform in contract with a precisions
required of 1 second</summary>
  <dtstart>20100524T220000Z</dtstart>
  <duration>T00:15</duration>
  <attach>http://scheduled.ws-calendar-service.com/contract1</attach>
  <related-to>
    <relationship>
      <uid>aaaaaaa2</uid><reltype>SF</reltype>
    </relationship>
```

```
  </related-to>
</VTODO>
<VTODO>
  <dtstamp></dtstamp>
  <uid>aaaaaaa2</uid>
  <description>second interval</description>
  <priority>high</priority>
  <summary>defines second behavior to perform in contract with a precision
required of 1 second</summary>
  <duration>T00:15</duration>
  <attach>http://scheduled.ws-calendar-service.com/contract2</attach>
  <related-to>
    <relationship>
      <uid>aaaaaaa1</uid><reltype>FS</reltype>
      <uid>aaaaaaa3</uid><reltype>SF+10m</reltype>
    </relationship>
  </related-to>
</VTODO>
<VTODO>
  <dtstamp></dtstamp>
  <uid>aaaaaaa3</uid>
  <description>second interval</description>
  <priority>high</priority>
  <summary>defines second behavior to perform in contract with a precision
required of 1 second</summary>
  <duration>30m</duration>
  <attach>http://scheduled.ws-calendar-service.com/contract2</attach>
  <related-to>
    <relationship>
      <uid>aaaaaaa2</uid><reltype>FS+T00:10</reltype>
    </relationship>
  </related-to>
</VTODO>
<components>
```

The shared performance component simplifies processing by establishing a common standard for all elements. Individual TODO elements need not be evaluated for performance.

A shared performance component is outside of the Sequence structure but within the components structure. If there is more than one Sequence in the components section, and if a Performance component is specified for the entire sequence, it specifies the performance characteristics for all Intervals in the components collection.

There SHALL be no more than one Performance component in the components collection. If there is a Performance component in the components collection, then any performance components within the individual intervals SHALL be ignored.

4.2.4 Partitions: Regular repeating sets

Perhaps the most common Sequence is one in which all Intervals have an identical duration, and each Interval follows immediately upon the completion of its predecessor. This partitioned set of intervals defines a Partition.

A partition is also degenerate in that the intervals are assumed to be identical in most aspects. For this reason, Description, Summary, and Priority are ignored when processing Partitions.

Example 10: Scheduled Partition with common contract

```
<components>
<RelatedComponent>
  <dtstamp></dtstamp>
  <uid>aaaaaaa0</uid>
  <summary>Defines performance requirements for entire sequence</summary>
  <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>
      <granularity>T00:00:02</dgranularity>
    </performance>
  </attach>
    <relationship>
      <uid>aaaaaaa1</uid><reltype>Parent</reltype>
    </relationship>
</RelatedComponent>
<VTODO>
  <dtstamp></dtstamp>
  <uid>aaaaaaa1</uid>
  <description>first contract</description>
  <dtstart>20100524T220000Z</dtstart>
  <duration>T00:15</duration>
  <attach>http://scheduled.ws-calendar-service.com/contract1</attach>
  <related-to>
    <relationship>
      <uid>aaaaaaa2</uid><reltype>SF</reltype>
    </relationship>
  </related-to>
</VTODO>
<VTODO>
  <dtstamp></dtstamp>
  <uid>aaaaaaa2</uid>
  <duration>T00:15</duration>
  <attach>http://scheduled.ws-calendar-service.com/contract2</attach>
  <related-to>
    <relationship>
      <uid>aaaaaaa1</uid><reltype>FS</reltype>
      <uid>aaaaaaa3</uid><reltype>SF+T00:10</reltype>
    </relationship>
  </related-to>
</VTODO>
<VTODO>
  <dtstamp></dtstamp>
  <uid>aaaaaaa3</uid>
  <duration>15m</duration>
  <attach>http://scheduled.ws-calendar-service.com/contract2</attach>
  <related-to>
    <relationship>
      <uid>aaaaaaa2</uid><reltype>FS+T00:10</reltype>
    </relationship>
  </related-to>
</VTODO>
<components>
```

Notice that the common contract is part of the Performance object. In this case, the Partition may start up to one minute early, and may not begin late. For each interval, the performance component that the duration be absolute.

Note also that Priority, Description, and Summary do not appear in any Interval in the Partition.

Note as well that this is a Scheduled Partition, and only the first Interval has a start time.

The same Scheduled Partition could be created with identical Durations for each Interval by specifying the Duration in the RelatedComponent.

Example 11: Scheduled Sequence with common contract, common duration

```
<components>
<RelatedComponent>
  <dtstamp></dtstamp>
```

```
  <uid>aaaaaaa0</uid>
  <summary>Defines performance requirements and duration for entire
sequence</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>
      <granularity>T00:00:02</dgranularity>
    </performance>
  </attach>
    <relationship>
      <uid>aaaaaaa1</uid><reltype>Parent</reltype>
    </relationship>
</RelatedComponent>
<VTODO>
  <dtstamp></dtstamp>
  <uid>aaaaaaa1</uid>
  <description>first contract</description>
  <dtstart>20100524T220000Z</dtstart>
  <attach>http://scheduled.ws-calendar-service.com/contract1</attach>
  <related-to>
    <relationship>
      <uid>aaaaaaa2</uid><reltype>SF</reltype>
    </relationship>
  </related-to>
</VTODO>
<VTODO>
  <dtstamp></dtstamp>
  <uid>aaaaaaa2</uid>
  <attach>http://scheduled.ws-calendar-service.com/contract2</attach>
  <related-to>
    <relationship>
      <uid>aaaaaaa1</uid><reltype>FS</reltype>
      <uid>aaaaaaa3</uid><reltype>SF+T00:10</reltype>
    </relationship>
  </related-to>
</VTODO>
<VTODO>
  <dtstamp></dtstamp>
  <uid>aaaaaaa3</uid>
  <attach>http://scheduled.ws-calendar-service.com/contract2</attach>
  <related-to>
    <relationship>
      <uid>aaaaaaa2</uid><reltype>FS+T00:10</reltype>
    </relationship>
  </related-to>
</VTODO>
<components>
```

Each Interval in this Sequence shares the 15 minute duration defined in the RelatedComponent as well as common Performance characteristics

4.3 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 invoked4.

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.

---

[4] From the RFC 5545 – see normative references

5 Calendar Service Interactions: Overview

This OASIS Committee has worked closely with the CalConnect TC-XML committee, which publishes its work through the IETF[5]. CalConnect is defining the core scheduling service interactions, i.e., scheduling an event, determining availability, etc., and publishing them as Cal-WS.

5.1 Glossary

5.2 Issues not addressed by this specification

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

6 Service Interactions: Protocol Overview

7 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.

8 Creating Calendar Resources

9 Retrieving Calendar Resources

10 Updating Calendar Resources

11 Deletion of Calendar Resources

12 Querying Calendar Resources

13 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 four.

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

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

All OASIS specifications require conformance

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 5, Calendar Service Interactions. The Technical Committee gratefully acknowledges their assistance and cooperation as well.

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