

# WS-Calendar Platform Independent Model (PIM) Version 1.0

## Committee Specification Draft 04

08 May 2015

### Specification URIs

#### This version:

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

#### Previous version:

<http://docs.oasis-open.org/ws-calendar/ws-calendar-pim/v1.0/cs01/ws-calendar-pim-v1.0-cs01.pdf> (Authoritative)  
<http://docs.oasis-open.org/ws-calendar/ws-calendar-pim/v1.0/cs01/ws-calendar-pim-v1.0-cs01.html>  
<http://docs.oasis-open.org/ws-calendar/ws-calendar-pim/v1.0/cs01/ws-calendar-pim-v1.0-cs01.doc>

#### Latest version:

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

### Technical Committee:

OASIS Web Services Calendar (WS-Calendar) TC

### Chair:

Toby Considine ([toby.considine@unc.edu](mailto:toby.considine@unc.edu)), University of North Carolina at Chapel Hill

### Editors:

William Cox ([wcox@coxsoftwarearchitects.com](mailto:wcox@coxsoftwarearchitects.com)), Individual  
Toby Considine ([toby.considine@unc.edu](mailto:toby.considine@unc.edu)), University of North Carolina at Chapel Hill

### Additional artifacts:

This prose specification is one component of a Work Product that also includes:

- XMI (UML in XML) documents representing the UML model described in the specification. XML is authoritative; EAP file is informative: <http://docs.oasis-open.org/ws-calendar/ws-calendar-pim/v1.0/csd04/xmi/>

### Related work:

This specification is related to:

- *WS-Calendar Version 1.0*. Edited by Toby Considine and Mike Douglass. 30 July 2011. Latest version: <http://docs.oasis-open.org/ws-calendar/ws-calendar/v1.0/ws-calendar-1.0-spec.html>.

**Abstract:**

The Platform Independent Model is an abstract model that defines conformance and improves interoperability of calendar and schedule models with each other and with WS-Calendar and Xcal, which are in turn based on IETF RFCs.

This is a Platform Independent Model under the Object Management Group's Model-Driven Architecture. The Platform Dependent Model to which this specification relates is the full model for WS-Calendar as expressed in XML (xCal).

The focus of this Platform Independent Model is on describing and passing schedule and interval information with information attachments.

**Status:**

This document was last revised or approved by the OASIS Web Services Calendar (WS-Calendar) TC on the above date. The level of approval is also listed above. Check the "Latest version" location noted above for possible later revisions of this document. Any other numbered Versions and other technical work produced by the Technical Committee (TC) are listed at [https://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=ws-calendar#technical](https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ws-calendar#technical).

TC members should send comments on this specification to the TC's email list. Others should send comments to the TC's public comment list, after subscribing to it by following the instructions at the "Send A Comment" button on the TC's web page at <https://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 TC's web page (<https://www.oasis-open.org/committees/ws-calendar/ipr.php>).

**Citation format:**

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

**[WS-Calendar-PIM-v1.0]**

*WS-Calendar Platform Independent Model (PIM) Version 1.0*. Edited by William Cox and Toby Considine. 08 May 2015. OASIS Committee Specification Draft 04. <http://docs.oasis-open.org/ws-calendar/ws-calendar-pim/v1.0/csd04/ws-calendar-pim-v1.0-csd04.html>. Latest version: <http://docs.oasis-open.org/ws-calendar/ws-calendar-pim/v1.0/ws-calendar-pim-v1.0.html>.

---

## Notices

Copyright © OASIS Open 2015. All Rights Reserved.

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

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

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

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

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

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

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

The name "OASIS" is a trademark of [OASIS](#), the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see <https://www.oasis-open.org/policies-guidelines/trademark> for above guidance.

---

# Table of Contents

<b>Table of Figures</b> .....	5
<b>List of Tables</b> .....	6
<b>1 Introduction</b> .....	7
1.1 Terminology .....	7
1.2 Normative References .....	7
1.3 Non-Normative References .....	7
1.4 Namespace .....	8
1.5 Naming Conventions .....	8
1.6 Editing Conventions .....	9
<b>2 Architectural Context [Non-Normative]</b> .....	10
2.1 Architectural Basis for the PIM .....	10
2.2 Standards for Representation of Time .....	10
2.3 Service-Oriented Architecture and the PIM .....	10
2.4 Model Driven Architecture .....	10
2.5 The PIM and the WS-Calendar PSM .....	11
2.6 Expression of the PIM UML Model .....	11
2.7 Structure of the PIM Model and Specification .....	12
<b>3 WS-Calendar PIM Terminology and Semantics</b> .....	13
3.1 Time Intervals and Collections of Time-Related Intervals .....	13
<b>4 The Platform-Independent Model</b> .....	17
4.1 Overview of the PIM .....	18
4.1.1 Model Diagram .....	18
4.1.2 Discussion .....	19
4.2 Classes for Date and Time, Duration, and Tolerance .....	19
4.2.1 Model Diagram .....	20
4.2.2 Discussion .....	21
4.2.3 Relationship to other PIM Components .....	21
4.3 The Interval Class .....	21
4.3.1 Model Diagram .....	21
4.3.2 Discussion .....	22
4.3.3 Relationship to other PIM Components .....	22
4.4 Payload Attachment to an Interval .....	22
4.4.1 Model Diagram .....	22
4.4.2 Discussion .....	23
4.4.3 Relationship to other PIM Components .....	23
4.5 The Gluon Class .....	23
4.5.1 Model Diagram .....	24
4.5.2 Discussion .....	24
4.5.3 Relationship to other PIM Components .....	25
4.6 Relationships among Gluons and Intervals .....	25
4.6.1 Model Diagram .....	26
4.6.2 Discussion .....	27
4.6.3 Relationship to other PIM Components .....	27

4.7	Recurrence and the PIM	27
4.8	Availability	27
4.8.1	Model Diagram	28
4.8.2	Discussion	28
4.8.3	Relationship to other PIM Components	28
5	Rules for WS-Calendar PIM and Referencing Specifications	29
5.1	Inheritance in WS-Calendar PIM	29
5.2	Covarying Elements	29
5.3	Specific Attribute Inheritance	29
6	Conformance	31
6.1	Conformance for Specifications Claiming Conformance to WS-Calendar PIM	31
6.2	General Conformance Issues (Non-Normative)	31
6.3	Conformance of Intervals	31
6.3.1	Intervals and Gluons	31
6.3.2	Other Attributes	31
6.4	Conformance of Bound Intervals and Sequences	32
6.5	Security Considerations (Non-Normative)	32
7	Examples using the PIM (Non-Normative)	33
7.1	Related Intervals	33
7.2	A Meeting Schedule	33
Appendix A.	Acknowledgments	36
Appendix B.	Revision History	37
Appendix C.	PIM to WS-Calendar PSM Transformation	39
C.1	General Transformations	39
C.2	Specific Transformations	39
C.2.1	Transformation for DateTime and Duration Types	39
C.2.2	Transformation for Tolerance Type	41
C.2.3	Transformation for Interval and Gluon Types	42
C.2.4	Transformation for Relationships	44
C.2.5	Transformation for Vavailability and FreeBusy	45
Appendix D.	PIM to IEC TC57 CIM Intervals and Sequences (Non-Normative Example)	47

## Table of Figures

Figure 4-1	The Complete WS-Calendar PIM UML Model	18
Figure 4-2	DateTimeType, DateType, TimeType, DurationType, and ToleranceType	20
Figure 4-3	IntervalType	21
Figure 4-4	Attaching a Payload to an Interval	22
Figure 4-5	Gluons, Intervals, and Relationship Links	24
Figure 4-6	Temporal Relationships	26
Figure 4-7	Temporal Relationship--startToFinish Negative 0.5 Gap	26
Figure 4-8	RelationLinkType and Relationship Types	26
Figure 4-9	Vavailability and Availability Recurrence Rules	28
Figure 7-1	PIM Expression of WS-Calendar Examples 3-05	33

Figure 7-2 Simple Meeting Schedule .....	35
Figure 7-3 PIM Source Classes for DateTimeType and Duration Types .....	40
Figure 7-4 WS-Calendar Target Classes .....	40
Figure 7-5 PIM Source Class for ToleranceType .....	41
Figure 7-6 WS-Calendar Target Classes for Tolerance Type .....	41
Figure 7-7 PIM IntervalType and GluonType .....	42
Figure 7-8 WS-Calendar Target IntervalType and GluonType .....	43
Figure 7-9 PIM RelationLinkType, LinkType, RelationshipType, and TemporalRelationshipType .....	44
Figure 7-10 PIM Vavailability Package Classes .....	46
Figure 7-11 Vavailability Package from iCalendar-availability-extension .....	46

## List of Tables

Table 3-1: Semantics: Foundational Elements .....	13
Table 3-2: Semantics: Relations, Limits, and Constraints .....	14
Table 3-3: Semantics: Inheritance .....	14
Table 3-4: Semantics: Describing Intervals .....	15
Table 7-1 PIM to PSM Mapping for DateTimeType and Duration Types .....	40
Table 7-2 PIM to PSM Mapping for ToleranceType .....	41
Table 7-3 PIM to PSM Mapping for IntervalType and GluonType .....	43
Table 7-4 PIM to PSM Mapping for Attributes of PIM RelationshipType and TemporalRelationshipType .....	45
Table 7-5 PIM to PSM Mapping for Enumeration Members .....	45

---

# 1 Introduction

All text is normative unless otherwise labeled. Notes and examples are non-normative; see Section 1.6 Editing Conventions.

## 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

- [ISO8601] ISO (International Organization for Standardization). Data elements and interchange formats -- Information interchange -- Representation of dates and times, Edition 3, 3 December 2004, (ISO 8601:2004)
- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, Uniform Resource Identifier (URI): Generic Syntax, STD 66, RFC 3986, January 2005.  
<http://www.ietf.org/rfc/rfc3986.txt>
- [RFC2119] Bradner, S., Key words for use in RFCs to Indicate Requirement Levels, <http://www.ietf.org/rfc/rfc2119.txt>, BCP 14, RFC 2119, March 1997.
- [RFC5545] Desruisseaux, B. Internet Calendaring and Scheduling Core Object Specification (iCalendar), <http://www.ietf.org/rfc/rfc5545.txt>, RFC 5545, September 2009
- [UML] OMG Unified Modeling Language (OMG UML), Infrastructure, Version 2.4.1, Object Management Group. <http://www.omg.org/spec/UML/2.4.1/Infrastructure> and OMG Unified Modeling Language (OMG UML), Superstructure, Version 2.4.1, <http://www.omg.org/spec/UML/2.4.1/Superstructure>, Object Management Group, August 2011
- [xCal] Daboo, C., Douglass, M., and S. Lees, xCal: The XML format for iCalendar, <http://tools.ietf.org/html/rfc6321>, IETF RFC 6321, August 2011.
- [XMI] MOF 2.0/XMI Mapping Specification, v2.1, September 2005, Object Management Group, <http://www.omg.org/spec/XMI/2.1/><sup>1</sup>

## 1.3 Non-Normative References

- [BPEL] Web Services Business Process Execution Language Version 2.0, 11 April 2007, OASIS Standard. <http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html>
- [BPMN] Business Process Model and Notation (BPMN) Version 2.0, Object Management Group, Version 2.0, <http://www.omg.org/spec/BPMN/2.0/>, January 2011
- [EnergyInterop-v1.0] Energy Interoperation Version 1.0. Edited by Toby Considine. 11 June 2014. OASIS Standard. <http://docs.oasis-open.org/energyinterop/ei/v1.0/os/energyinterop-v1.0-os.html>. Latest version: <http://docs.oasis-open.org/energyinterop/ei/v1.0/energyinterop-v1.0.html>. PDF is authoritative.
- [Enterprise Architect] Sparx Enterprise Architect 10.0, used to produce [UML] 2.4.1 diagrams, EAP and [XMI] version 2.1 files, <http://sparxsystems.com/>.
- [IANA] The Internet Assigned Numbers Authority, <http://www.iana.org>.

---

<sup>1</sup> The UML tools used by the TC support version 2.1, which is not the most recent as of this date.

41	<b>[IEC CIM]</b>	IEC 61968/61970, International Electrotechnical Commission, collection of specifications, various dates, <a href="http://www.iec.ch">http://www.iec.ch</a> <sup>2</sup>
42		
43	<b>[MDA-Overview]</b>	The Architecture of Choice for a Changing World, Object Management Group, <a href="http://www.omg.org/mda/">http://www.omg.org/mda/</a>
44		
45	<b>[MDA]</b>	OMG Model Driven Architecture Specifications, Object Management Group, <a href="http://www.omg.org/mda/specs.htm">http://www.omg.org/mda/specs.htm</a>
46		
47	<b>[PIM Examples]</b>	Examples for WS-Calendar Platform-Independent Model (PIM) Version 1.0, OASIS Committee Technical Note, in progress.
48		
49	<b>[Relationships]</b>	M. Douglass, Support for Icalendar Relationships, <a href="http://tools.ietf.org/html/draft-douglass-ical-relations-02">http://tools.ietf.org/html/draft-douglass-ical-relations-02</a> , IETF Internet Draft Version 02, January 7, 2014
50		
51	<b>[SOA-RAF]</b>	Reference Architecture Foundation for Service Oriented Architecture Version 1.0, 04 December 2013. OASIS Committee Specification. <a href="http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/cs01/soa-ra-v1.0-cs01.html">http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/cs01/soa-ra-v1.0-cs01.html</a> PDF is authoritative.
52		
53		
54	<b>[SOA-RM]</b>	OASIS Reference Model for Service Oriented Architecture 1.0, October 2006. OASIS Standard. <a href="http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.html">http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.html</a>
55		
56	<b>[Availability]</b>	C. Daboo, M. Douglass, Calendar Availability, <a href="https://datatracker.ietf.org/doc/draft-ietf-calext-availability/">https://datatracker.ietf.org/doc/draft-ietf-calext-availability/</a> , IETF Internet Draft Version 00, 23 March, 2015.
57		
58		
59	<b>WS-Calendar]</b>	WS-Calendar Version 1.0. Edited by Toby Considine and Mike Douglass. 30 July 2011, OASIS Committee Specification. <a href="http://docs.oasis-open.org/ws-calendar/ws-calendar-spec/v1.0/cs01/ws-calendar-spec-v1.0-cs01.html">http://docs.oasis-open.org/ws-calendar/ws-calendar-spec/v1.0/cs01/ws-calendar-spec-v1.0-cs01.html</a> (PDF is authoritative)
60		
61		
62		
63	<b>[XMLSchema]</b>	W3C XML Schema Definition Language (XSD) 1.1, World Wide Web Consortium, Part 1: Structures, S. Gao, C. M. Sperberg-McQueen, H. S. Thompson, N. Mendelsohn, D. Beech, M. Maloney, Editors, W3C Recommendation, 5 April 2012, <a href="http://www.w3.org/TR/2012/REC-xmlschema11-1-20120405/">http://www.w3.org/TR/2012/REC-xmlschema11-1-20120405/</a> . Latest version available at <a href="http://www.w3.org/TR/xmlschema11-1/">http://www.w3.org/TR/xmlschema11-1/</a> .
64		
65		
66		
67		
68		
69		
70		
71		

## 72 1.4 Namespace

73 There are no XML namespaces defined in this specification.

## 74 1.5 Naming Conventions

75 This specification follows a set of naming conventions for artifacts defined by the specification, as follows:

76 For the names of attributes in UML classes the names follow the lower camelCase convention, with all  
77 names starting with a lower case letter. For example, an attribute name might be

78 `temporalRelationship`

79 The names of UML classes follow the upper CamelCase convention with all names starting with an Upper  
80 case letter followed by "Type".

81 `TemporalRelationshipType`

---

<sup>2</sup> In this specification, the relevant parts are *IEC 61968-9, Edition 2.0, October, 2013*, [http://webstore.iec.ch/webstore/webstore.nsf/ArtNum\\_PK/48719?OpenDocument](http://webstore.iec.ch/webstore/webstore.nsf/ArtNum_PK/48719?OpenDocument) and *IEC 61970-301, Edition 5.0, December 2013*, [http://webstore.iec.ch/webstore/webstore.nsf/ArtNum\\_PK/49080?OpenDocument](http://webstore.iec.ch/webstore/webstore.nsf/ArtNum_PK/49080?OpenDocument)



82 The UML Primitive Type *String* [UML, *Infrastructure*]<sup>3</sup> is used in this specification.

## 83 1.6 Editing Conventions

84 For readability, UML attribute names in tables appear as separate words. The actual names are  
85 lowerCamelCase, as specified above, and do not contain spaces.

86 Attribute and type names are usually in an *italic* face.

87 All items in the tables not marked as “optional” are mandatory.

88 Information in the “Specification” column of tables is normative. Information appearing in the “Note”  
89 column is non-normative.

90 Text indicated as “Note” are non-normative.

91 All sections explicitly described as examples are non-normative.

92 All examples with gray highlight are non-normative.

93 All Appendices are non-normative.

94 In [UML] diagrams, purple background is used for an abstract class.

---

<sup>3</sup> See <http://www.omg.org/spec/UML/20110701/PrimitiveTypes.xmi>

---

## 95 2 Architectural Context [Non-Normative]

96 In this section we discuss the context in which this specification was developed, its purpose, and selected  
97 applications.

### 98 2.1 Architectural Basis for the PIM

99 The PIM is defined as a more abstract model for describing and communicating schedules as defined in  
100 **[WS-Calendar]**, **[EMIX]**, **[EnergyInterop-v1.0]**, **[OBIX]**, and **[SPC201]**, among many others. This  
101 expression uses typical ways of expressing schedule, linked lists, directed graphs, and is consistent with  
102 algorithms for graph, list, and schedule management.

103 In summary, there are several anticipated architectural benefits of the PIM:

- 104 1. Expression of schedules in a common manner showing temporal structures and taking advantage  
105 of differing views of a single schedule
- 106 2. Relocatable subroutines that may be used dynamically at run time
- 107 3. Automatable transformations between the abstract and concrete schedules in the PIM and WS-  
108 Calendar respectively
- 109 4. Broader use of scheduling concepts in other domains and PSMs allowing automatable  
110 transformations across other domains

111 Schedule and values attached to time intervals in schedule are fundamental to planning and carrying out  
112 operations in most domains. The WS-Calendar PIM provides a common model for expressing and  
113 managing such schedules.

### 114 2.2 Standards for Representation of Time

115 We rely on **[ISO8601]** for description of date, time, and duration. Many of the concepts in that standard  
116 are well known to users of iCalendar **[RFC5545]** and XML Schema **[XMLSchema]**, both of which share  
117 similar but slightly different subsets of the expressive power of **[ISO8601]**. For example, we define a  
118 conformed string for an attribute called *ISO8601Duration* which differs in detail from the perhaps more  
119 familiar XML Schema and iCalendar.

120 PSMs may restrict or profile time expressions in the PIM. For example, many industrial control systems  
121 define time intervals with start and end time, which is a conformant 8601 definition. For purposes of  
122 relocatable schedules, as used in e.g. **[EMIX]** and **[EnergyInterop-v1.0]** this PIM uses start time and  
123 duration only, another conformant 8601 definition.

### 124 2.3 Service-Oriented Architecture and the PIM

125 WS-Calendar PIM is an information model that may be used to define service request and response  
126 message payloads. For that purpose it assumes a background of definitions and of roles, names, and  
127 interaction patterns. Non-normative examples may use terminology defined in the OASIS Standard  
128 *Reference Model for Service Oriented Architecture* **[SOA-RM]**.

129 Service-Oriented Architecture comprises not only the services and interaction patterns, but also the  
130 information models that support those services and make the actions meaningful. The WS-Calendar PIM  
131 is such an information model for expressing schedule and time related information in a consistent manner  
132 and to permit easy transformation or adaptation into IETF iCalendar related specifications and among  
133 Platform-Specific Models based on this PIM.

### 134 2.4 Model Driven Architecture

135 The Object Management Group's Model Driven Architecture **[MDA-Overview]****[MDA]** provides a  
136 framework to describe relationships between Unified Modeling Language **[UML]** models.

137 An instance of MDA has two classes of models:

- 138 • A single *Platform-Independent Model*, abbreviated *PIM* (pronounced as spelled)
- 139 • One or more *Platform-Specific Models*, abbreviated *PSM* (pronounced as if spelled *pism*)

140 The PIM typically captures the more abstract relationships, clarifying the architecture. Each PSM is bound  
141 to a particular *platform*.

142 The art of establishing an MDA includes defining platforms and a PIM and PSMs, to solve interesting  
143 important and useful problems. Artifacts expressed in different PSMs may more readily be exchanged  
144 and understood with reference to the related PIM, making interoperation simpler and semantics more free  
145 from irrelevant detail.

## 146 2.5 The PIM and the WS-Calendar PSM

147 In this specification we define a PIM or Platform-Independent Model with respect to which the **[WS-  
148 Calendar]** specification may be treated as a PSM or Platform-Specific Model; the platform may be  
149 considered to be iCalendar **[RFC5545]**, **[xCal]**, and **[Vavailability]**.

150 We use “the PIM” to mean “the WS-Calendar PIM” in this specification.

151 **[iCalendar]** uses a set of definitions and a platform, developed over many years and much use, to  
152 express relationships, times, events, and availability. The expression is very simple, but in the aggregate  
153 relatively complex and less suitable to UML expression—the several key types (components) have sets of  
154 values, types, and parameters associated with them in a relatively flat hierarchy.

155 This PIM addresses the key **[WS-Calendar]** abstractions in a manner that allows for a better  
156 understanding of the nature and information model for those abstractions. Our purpose is to create a  
157 more abstract model of the key concepts in WS-Calendar for easier use in application development,  
158 standardization, and interoperation. As such, this PIM does not normatively reference any PSM, including  
159 but not limited to **[WS-Calendar]**.

160 The MDA presumes transformations from UML models to UML models. The UML model for **[WS-  
161 Calendar]** is structured very differently from that of the PIM. We describe the transformation in detail in  
162 non-normative Appendix C.

163 This specification does not rely on any specific MDA tooling or environments to be useful.

## 164 2.6 Expression of the PIM UML Model

165 The PIM is a **[UML]** model. We represent the PIM as a normative **[XMI]** serialization of the PIM UML  
166 model. The model itself is described using **[Enterprise Architect]**; an Enterprise Architect Project file is  
167 part of this work product but is non-normative. Many modeling tools use XML serialization for model  
168 exchange.

169 The terminology for attributes of an object, and how to describe an object or type differs between  
170 **[XMLSchema]** and **[UML]**. Attributes of a class in UML that is expressed in standards mappings to XML  
171 Schema are called either attributes (expressed in *name=value* format in XML) or elements. Since this  
172 specification is based on UML, we use the term *attribute* throughout.<sup>4</sup>

173 The PIM model is constrained, and by applying semantic rules the model allows succinctly described  
174 relocatable graphs of Intervals

175 For example, an instance of *IntervalType* (see Figure 4-3 IntervalType) might have only *duration*; the PIM,  
176 however, describes *duration* as optional (cardinality 0..1). Rules in this specification show how a specific  
177 representation is to be interpreted, typically by inheriting values from elsewhere. Conceptually, the actual  
178 values depend on the context and applied rules.

179 An Interval notionally has a start time, but that also is optional in the PIM. Finally, an Interval does not  
180 have an end time (expressed in Figure 4-3 as dtEnd of cardinality 0. We keep the dtEnd attribute for ease

---

<sup>4</sup> There are UML stereotypes to express the nature of an XML Schema export, indicating whether a UML attribute should be represented as an XSDattribute or XSDelement.

181 of use in PSMs and for intermediate stages of mapping into the canonical start and duration model, as  
182 well as mapping into and from models that define intervals with all three of start, end, and duration.  
183 These characteristics are as defined in **[WS-Calendar]** and describe an abstract Interval with at most a  
184 start time and duration. This is in contrast to some historical models that require each interval to contain a  
185 start and end time, or occasionally start, end, and duration. The added flexibility of relocatable sets or  
186 schedules comprised of Intervals and Gluons makes the expression of such a relocatable schedule easy  
187 and reusable, thus permitting a powerful abstraction to be applied to all sorts of scheduling expressions.  
188 In addition the mapping capability to and from the PIM allows interoperation with systems with less  
189 conveniently relocatable intervals.

## 190 **2.7 Structure of the PIM Model and Specification**

191 The PIM consists of a small number of key classes with a sub-package for the Availability **[Vavailability]**  
192 abstractions.<sup>5</sup> We have not otherwise subdivided the core model, but expect that conforming  
193 specifications and implementations may claim conformance to sub-parts of the PIM, e.g. to only the  
194 Interval.

195 We encourage use of the entire PIM, but understand that some aspects of the abstract model may be  
196 more complex than needed to address specific problems. We consider such profiles of the PIM to  
197 themselves be Platform-Specific Models.

198 We generally take the names for abstractions in the PIM from the names in **[WS-Calendar]** to simplify  
199 implementations and mappings.

200 Many values in the XML Serialization **[xCAL]** of iCalendar are conformed strings, that is, strings that meet  
201 specific defined patterns. We require similar standardized formats for conformed strings, and record the  
202 type in the PIM using the UML primitive type *String*. This allows easy transformation between this PIM  
203 and the PSMs. We include references to **[ISO8601]** and other specifications in the comments in the  
204 model.

---

<sup>5</sup> Note: The Vavailability definition is in process in the IETF.

## 205 3 WS-Calendar PIM Terminology and Semantics

206 WS-Calendar PIM semantics are defined in this section. The terminology aligns closely with that is **[WS-**  
207 **Calendar]**.

208 Note: This specification and **[WS-Calendar]** share the same semantics and terminology, which allows  
209 easier exchange of information across execution environments as well as consistency across Platform  
210 Specific Models related to this specification.

211 The normative definitions of terms are included here in Section 3.

### 212 3.1 Time Intervals and Collections of Time-Related Intervals

213 We begin with specialized terminology for the segments of time, and for groups of related segments of  
214 time. These terms are defined in Table 3-1 through Table 3-4 below.

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

Time Segment	Definition
Duration	Duration is the length of a time interval. In the PIM the value set from <b>[ISO8601]</b> is used; informally there are several additional representations for duration in the PIM compared to either <b>[xCal]</b> or <b>[XMLSchema]</b> but all those representations are included. See Section 4.2.
Interval	An Interval has as attributes a single Duration derived from <b>[ISO8601]</b> . An Interval may be part of a Sequence. An entire Sequence can be scheduled by scheduling a single Interval in that sequence. For this reason, Intervals are defined through Duration rather than through dtStart or dtEnd.
Sequence	A Sequence is a set of Intervals with defined temporal relationships. Sequences may have gaps between Intervals, or may be in parallel or overlapping. A Sequence is re-locatable, i.e., it does not have a specific date and time at which it starts or finishes. A Sequence may consist of a single Interval. A Sequence may optionally include a Lineage.  A Sequence CAN be scheduled or applied multiple times through repeated reference by different Gluons that give specific start time to the Sequence.
Partition	A Partition is a set of consecutive Intervals without gaps or overlap among them. The Partition includes the trivial case of a single Interval. Partitions MAY be used to define a single service or value set that varies over time (a time series). Examples include energy prices over time and energy usage over time.
Gluon	A Gluon influences the serialization of Intervals in a Sequence, though inheritance and through schedule setting. The Gluon is similar to the Interval, but has no effect beyond that of a reference until the Gluon is applied to a referenced Interval or Sequence.
Artifact	An Artifact is the information attached to, and presumably that occurs during or is relevant to the associated Interval. The Artifact is a placeholder. The contents of the Artifact are not specified here; rather the Artifact is an abstract type <b>[UML]</b> that provides an extension base. Artifacts MAY inherit elements as do Intervals within a Sequence. A Conforming specification MUST describe where and why its inheritance rules differ from those in this specification.

216 The PIM works with groups of Intervals that have relationships between them. These relations constrain  
217 the final description for a schedule or a schedule-based service. Relationships can control the ordering of

- 218 Intervals in a Sequence. They can describe when a service can be, or is prevented from, being invoked.  
 219 They establish the parameters for how information will be shared between elements using Inheritance.  
 220 The terminology for these relationships is defined in Table 3-2.  
 221 *Table 3-2: Semantics: Relations, Limits, and Constraints*

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

- 222 WS-Calendar describes how to modify and complete the specification of Sequences. WS-Calendar calls  
 223 this process Inheritance and specifies a number of rules that govern inheritance. Table 3-3 defines the  
 224 terms used to describe inheritance, with rewording to address this PIM.  
 225 *Table 3-3: Semantics: Inheritance*

Term	Definition
Lineage	The ordered set of Parents that results in a given inheritance or execution context for a Sequence.
Inheritance	Parents bequeath information to Children that inherit them. If a child does not already possess that information, then it accepts the inheritance. WS-Calendar specifies rules whereby information specified in one informational object is considered present in another that is itself lacking expression of that information. This information is termed the Inheritance of that object.
Bequeath	A Parent Bequeaths attributes (Inheritance) to its Children.
Inherit	A Child Inherits attributes (Inheritance) from its Parent.

Term	Definition
Covarying Attributes	Some attributes are inherited as a group. If any member of that group is expressed in a Child, all members of that group are deemed expressed in that Child, albeit some may be default values. These characteristics are called covarying or covariant. A parent bequeaths covarying characteristics as a group and a child accepts or refuses them as a group.
Decouplable Attributes	Antonym for Covarying Attributes. Decouplable Attributes can be inherited separately.

226 As Intervals are processed, as Intervals are assembled, and as inheritance is processed, the information  
227 conveyed about each element changes. When WS-Calendar is used to describe a business process or  
228 service, it may pass through several stages in which the information is not yet complete or actionable, but  
229 is still a conforming expression of time and Sequence. Table 3-4 defines the terms used when discussing  
230 the processing or processability of Intervals and Sequences.

231 During the life cycle of communications concerning Intervals, different information may be available or  
232 required. For service performance, Start Duration and the Attachment Payload must be complete. These  
233 may not be available or required during service advertisement or other pre-execution processes. Table  
234 3-4 defines the language used to discuss how the information in an Interval is completed.

235 *Table 3-4: Semantics: Describing Intervals*

Term	Definition
Designated Interval	An Interval that is referenced by a Gluon is the Designated Interval for a Series. An Interval can be Designated and still not Anchored.
Anchored	An Interval is Anchored when it includes a Start or End, either directly or through Binding. A Sequence is Anchored when its Designated Interval is Anchored.
Unanchored	An Interval is Unanchored when it includes neither a Start nor an End, either internally, or through Binding. A Sequence is Unanchored if its Designated Interval is Unanchored. <i>Note: a Sequence that is re-used may be Unanchored in one context even while it is Anchored in another.</i>
Binding	Binding is the application of information to an Interval or Gluon, information derived through Inheritance or through Temporal Assignment.
Bound Attribute	A Bound Attribute refers to an Attribute and its Value after Binding, e.g., a Bound Duration.
Bound Interval	A Bound Interval refers to an Interval and the values of its Elements after Binding.
Bound Sequence	A Bound Sequence refers to a Sequence and the values of its Intervals after Binding.
Partially Bound	Partially Bound refers to an Interval or a Sequence which is not yet complete following Binding, i.e., the processes cannot yet be executed.
Fully Bound	Fully Bound refers to an Interval or Sequence that is complete after Binding, i.e., the process can be unambiguously executed when Anchored.
Unbound	An Unbound Interval or Sequence is not complete, and must receive inheritance to be fully specified. A Sequence or Partition is Unbound if it contains at least one Interval that is Unbound.

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



237

---

## 4 The Platform-Independent Model

238

In this section we define the PIM.

239

Each subsection has an introduction, a diagram, and discussion of the relationship of the components to the rest of the PIM.

240

241

This Platform-Independent Model (PIM) **[MDA]** describes an abstraction from which the Platform-Specific Model (PSM) of **[WS-Calendar]** and other models can be derived. The intent is twofold:

242

243

(1) To define an abstraction for calendar and schedule more in the style of web services descriptions, which may be used directly, and

244

245

(2) To define the PIM as a model allowing easy transformation or adaptation between systems using the family of WS-Calendar specifications (such as [WS-Calendar], [xCal], [iCalendar]) as well as those addressing concepts of time intervals and Sequences (such as **[IEC CIM]**, **[EnergyInterop-v1.0]**, and **[EMIX]**).

246

247

248

249

The following subsections each contain a description of the relevant portions of the model, addressing in turn

250

251

- Section 4.1 Overview of the PIM

252

- Section 4.2 Classes for Date and Time, Duration, and Tolerance

253

- Section 4.3 The Interval Class

254

- Section 4.4 Payload Attachment to an Interval

255

- Section 4.5 The Gluon Class

256

- Section 4.6 Relationships among Gluons and Intervals

257

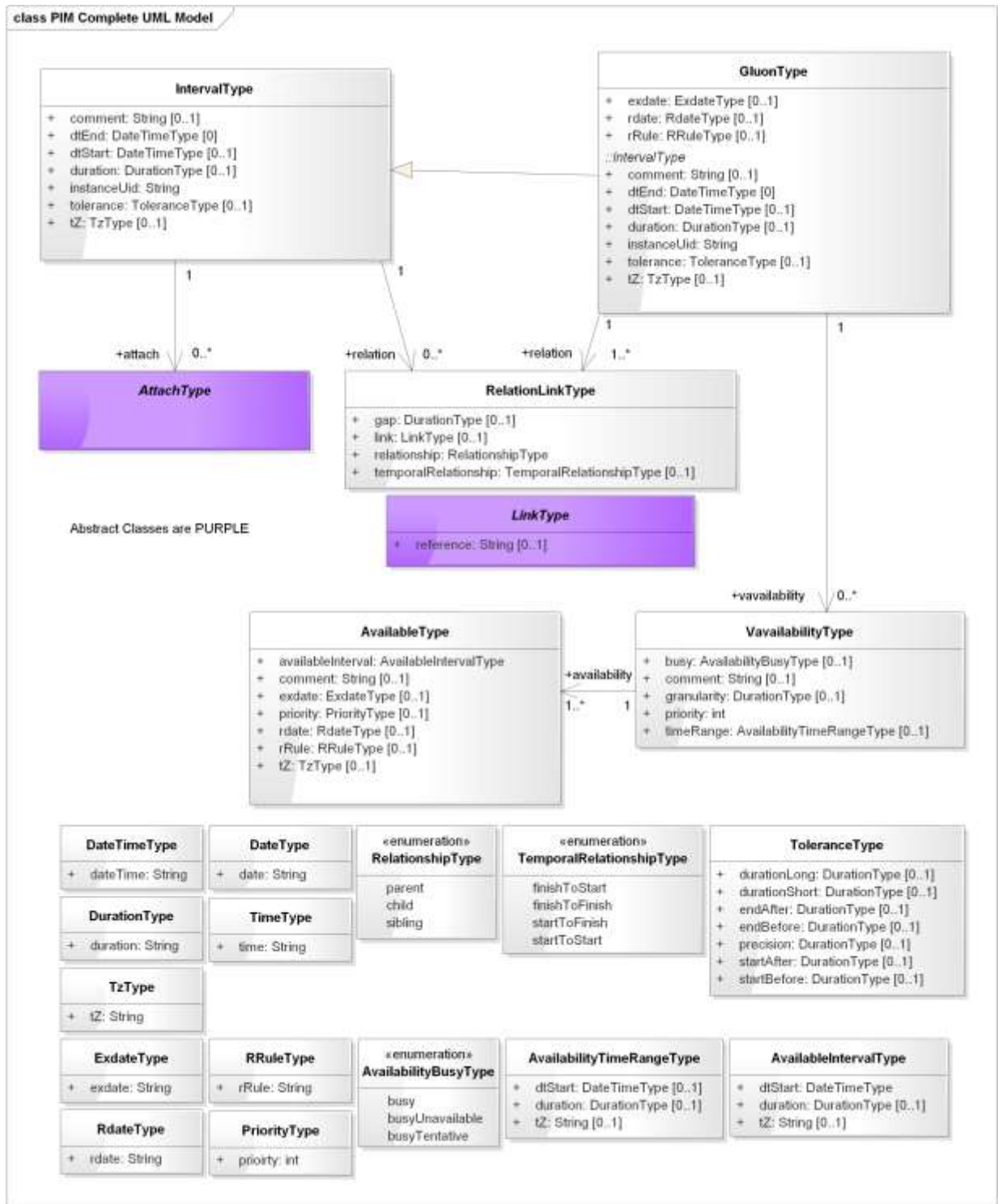
- Section 4.7 Recurrence and the PIM

258

- Section 4.8 Availability

259 **4.1 Overview of the PIM**

260 **4.1.1 Model Diagram**



261  
262 *Figure 4-1 The Complete WS-Calendar PIM UML Model. Abstract classes have violet background. Classes changed*  
263 *since WD13 have yellow background.*

## 264 4.1.2 Discussion

265 Primitive types in the PIM express fundamental information related to date, time, and duration, follow  
266 **[RFC5545]** **[ISO8601]** **[Vavailability]** and are a superset of those expressed in **[iCalendar]**. Many are  
267 conformed versions of the **[UML]** primitive type *String*.

268 Associations in the PIM are directional, but profiles and PSMs derived or derivable from the PIM MAY  
269 have non-directional associations, or vary the direction of associations to fit their particular platform(s)  
270 and purposes.

271 Note: non-directional associations present a barrier to serializability; we RECOMMEND that PSMs  
272 typically would use directional associations unless their purpose is to derive further PSMs.

273 The cardinality for all attributes and associations is specified in the PIM. Profiles and PSMs with respect  
274 to the PIM MAY have different cardinality.

275 Attachments are made via the abstract class *AttachType* as described in Section 4.4.

276 We have used the **[RFC5545]** and **[ISO8601]** and **[Vavailability]** attribute, type, parameter, and value  
277 names wherever possible for ease of mapping to and from that terminology.

278 Per **[ISO8601]** a fully bound Interval can be described by any two of

- 279 • *dtStart*—the date & time for the start of the Interval
- 280 • *dtEnd*—the date & time for the end of the Interval
- 281 • *duration*—the duration of the interval

282 In the PIM UML model, the three key values for an interval, only two of which are required in fully bound  
283 Intervals, are each optional. This permits a conforming PSM to have zero or more of the three key values.  
284 The PIM generally requires that at most *dtStart* and *duration* are used to allow relocatable schedules.

285 The Rules in Section 5 describe how information for a bound interval is determined. *GlucnType* is a  
286 subclass of *IntervalType* but has a more restrictive cardinality for *dtEnd* and for *relation*.

287 Classes related to **[Vavailability]** follow the semantics and cardinality in that specification.

288 NOTE: The referenced specification is an Internet Draft and is expected to be updated with a Standards  
289 Track IETF RFC in the near future.

## 290 4.2 Classes for Date and Time, Duration, and Tolerance

291 In this section we introduce key concepts and expressions for time including

- 292 • *DateTime*
- 293 • *DurationType*
- 294 • *ToleranceType*
- 295 • *DateType*
- 296 • *TimeType*

297 Relationships are described in Section 4.5.

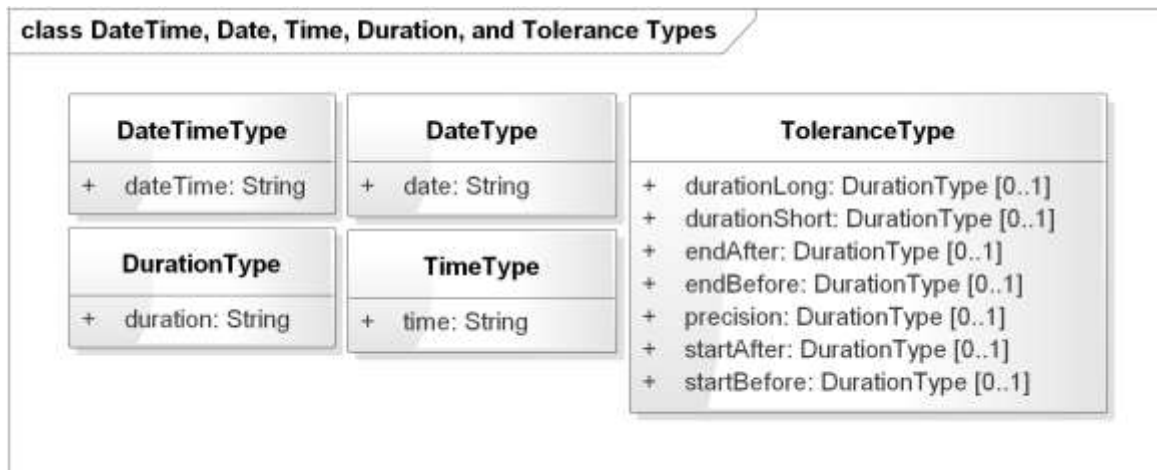
298 No timing of events, whether descriptive or prescriptive, can be perfectly accurate within the limits of  
299 measurement of real systems. Tolerance is an optional attribute that applies to the duration, allowing full  
300 flexibility in the description of permissible or expected variation in duration.

301 The containing Interval might start early or late, end early or late, or have a duration that may be short or  
302 long with respect to the nominal value. The *precision* in *ToleranceType* is a *DurationType* that expresses  
303 the precision for tolerances.<sup>6</sup>

---

<sup>6</sup> This differs from *granularity* in *VavailabilityType*, which describes the availability interval length.

304 **4.2.1 Model Diagram**



305  
306 *Figure 4-2 DateTimeType, DateType, TimeType, DurationType, and ToleranceType*

307 All DateTime, Date, Time, and Duration values are expressed as conformed strings, that is, the type is  
308 *String* and the content of the string determines respectively the date, time, and the duration.

309 The values of the following SHALL be expressed as conformed strings as described in the normative  
310 reference **[RFC5545]** and as otherwise indicated below. The optional sign for *DurationValueType* MUST  
311 be available in PIM conformed strings for *DurationType*, and the allowable patterns excluding sign MUST  
312 conform to **[ISO8601]**. Other than conformed string grammar, the references are as follows:

- 313 • DateTime (Section 3.3.5, Date-Time)
- 314 • DurationValueType (Section 3.3.6, Duration)
- 315 • Date (Section 3.3.4, Date)
- 316 • Time (**[ISO8601]** Section 4.2, Time of Day, and **[XMLSchema]** Part 2, Section 3.2.8)

317 Conforming PSMs MUST describe the semantics applied to DateType and TimeType including the  
318 application of Time Zones.

319 NOTE:**[XML Schema]** has Date, Time, DateTime, and Duration as basic types but not all values (e.g. for  
320 Duration) are expressed. Likewise, **[RFC5545]** defines Date, DateTime, and Duration

321 The class *ToleranceType* is comprised of a set of optional attributes of *DurationType*. Tolerances can be  
322 expressed in any combination.

323 The String values for any attribute of *ToleranceType* SHALL be non-negative or a minus sign SHALL be  
324 ignored.

325 A PSM SHALL state rules for non-negative *ToleranceType* attributes in their conformance statement.

326 PSMs SHOULD specify that the cardinality of *tolerance* MUST be zero if *tolerance* is empty.

327 Note: The complexity of rules addressing the relationships of tolerances in start, end, and duration will  
328 likely lead to implementation-specific rules limiting the concurrent uses of tolerance attributes.

329 It is RECOMMENDED that a PSM include consistency requirements and limitations on the attributes of  
330 *ToleranceType* that might be used. It is RECOMMENDED that profiled sets of tolerances be specified by  
331 a PSM. PSMs MUST document in their conformance statement any consistency requirements, limitations  
332 on, and profiled sets of tolerances.

333 For example, *startAfter* = PT5M and *startBefore* = PT10M indicates that the associated action or the  
334 interval to which *AttachType* applies may start in the range from ten minutes before the indicated *dtStart*  
335 to five minutes after the indicated *dtStart*.

336 Tolerances can allow (e.g.) randomization of intervals to ensure that certain activities do not occur  
337 “simultaneously.” Continuing the example, additional deployment semantics for randomization might  
338 apply to that 15-minute interval.

339 PSMs MAY include assumptions or explicit statement of e.g. probability density functions or other  
340 indications of expected behavior. Such assumptions and/or explicit statements SHALL be included in the  
341 conformance statement.

342 Tolerances also express information about schedules that enables the application of optimization  
343 techniques both across and within schedules.

## 344 4.2.2 Discussion

345 These concepts are based on [ISO8601] and are as expressed in [iCalendar] as conformed strings. It is  
346 important to note that *DurationType* is identical to *neither* the XML Schema Specification [XMLSchema]  
347 *Duration*<sup>7</sup> nor the [xCal] and [iCalendar] specification for duration.

348 PSMs MAY express DurationType differently; if so the differences MUST be described in their  
349 conformance statement.

## 350 4.2.3 Relationship to other PIM Components

351 These concepts are pervasive in the WS-Calendar PIM. The fundamental understanding of time and  
352 duration must be consistent and identical to that in [iCalendar] for clean interoperability and  
353 transformation. Documentation of any differences in expression MUST be included in the conformance  
354 statement for any PSM claiming conformance to this PIM. Moreover, a mapping MUST be provided both  
355 directions between the types defined here and those in a PSM claiming conformance.

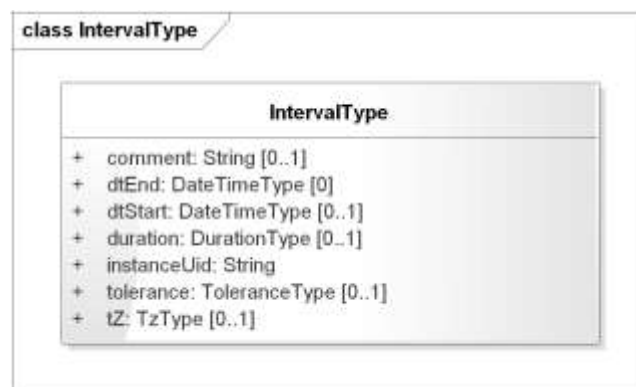
## 356 4.3 The Interval Class

357 The Interval is fundamental—a bound interval starts at a particular time, runs for a specific duration, and  
358 ends at a particular time. This is reflected in Figure 4-3.

359 But there are many possible standards-based expressions of a time interval, and significant differences in  
360 relocatability of schedules including Intervals depending on choices made in representation.

361 We describe the PIM representation in this section.

### 362 4.3.1 Model Diagram



363  
364 *Figure 4-3 IntervalType*

---

<sup>7</sup> While [iCalendar], [WS-Calendar], and this PIM conform to [ISO8601], only this PIM requires all standard notations from 8601 as well as sign for duration. Likewise [XMLSchema] does not include the full specification in [ISO8601]. There are duration strings included in [XMLSchema] that are not in [iCalendar] and *vice versa*.

365 **4.3.2 Discussion**

366 Class *IntervalType* is the model for a time interval; while logically any two or three of the set {*dtStart*,  
367 *dtEnd*, and *duration*} can express an interval, there are significant advantages to adopting a single  
368 canonical form, particularly one where the semantics are cleanly expressed. Intervals may be, and are,  
369 expressed many ways [ISO8601] section 4.4 This PIM requires a specific expression that optionally  
370 includes start time and duration but not end time.

371 Individual PSMs may use different expressions, but SHOULD recognize in their design that relocation and  
372 scheduling of sets of intervals is a very common operation; as we will show later, an entire schedule of  
373 Intervals in this WS-Calendar PIM can be scheduled with a single operation, whereas in other  
374 representations each *dtStart* and *dtEnd* might have to be modified when scheduling.

375 PSMs SHALL describe their requirements and restrictions on Interval descriptions in their conformance  
376 statements.

377 **4.3.3 Relationship to other PIM Components**

378 The information in the *IntervalType* class is fundamental to expression of time interval. [ISO8601].

379 To maintain temporal structure while allowing correlated values, payload values are attached to an  
380 Interval (or its subclass *GlionType*), as described in the next section.

381 **4.4 Payload Attachment to an Interval**

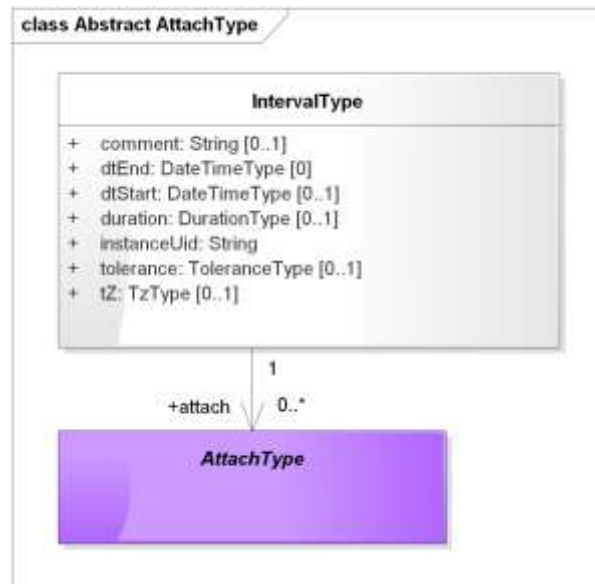
382 A payload, which may be comprised of multiple subparts within a single class, or a reference, is attached  
383 to an Interval. This differs from other approaches that have been taken, such as

384 (a) A class containing a value as well as a description of a relevant Interval (e.g. a measurement that  
385 applies to an included Interval)

386 (b) Associating a particular measurement to an interval (the association is the wrong direction)

387 The association is directional, and must be present for use of an Interval object in a concrete way.

388 **4.4.1 Model Diagram**



389

390 *Figure 4-4 Attaching a Payload to an Interval*

## 391 4.4.2 Discussion

392 **[WS-Calendar]** (line 219) requires that the Attachment Payload and Start Duration must be complete for  
393 service performance. In contrast, the PIM defines the cardinality of *attach* to be 0..\* to allow for abstract  
394 schedules, including those to which payloads are bound before use. This mirrors the manner in which  
395 attribute values are inherited by Intervals during Binding.

396 A PSM claiming conformance to this PIM SHALL document in its conformance statement any changes in  
397 the definition of *AttachType* and/or the cardinality of associations used for payload attachment.

## 398 4.4.3 Relationship to other PIM Components

399 The *IntervalType* is fundamental; application information is attached to objects of class *IntervalType* by a  
400 clear, directional association. This makes the temporal structure of schedules independent of associated  
401 information, and of the nature of the associated information by judicious definition of concrete (non-  
402 abstract) attachment types.

## 403 4.5 The Gluon Class

404 A Gluon may be thought of as a reference to a Sequence (a set of temporally-related intervals), with the  
405 same attributes as an Interval for simplicity of inheritance.

406 A sequence MAY be referenced by zero or more gluons; the view of a sequence and the values as  
407 applied by the Rules in Section 5 are determined by attribute values in the referencing Gluon and values  
408 that may be inherited from the referencing gluon such as start time and duration.

409 More formally, a *Gluon* references schedules comprised of temporally related Intervals and Gluons, while  
410 providing that logical information such as the duration of Interval objects may be stated explicitly or be  
411 determined by inheritance from the respective Lineages.

412 The structure defined enables the creation of directed graphs of Interval objects with reuse of  
413 components. Those sub graphs may therefore act as reusable sub-schedules, or considered as sub-  
414 routines. See Section 7 Examples using the PIM (Non-Normative).

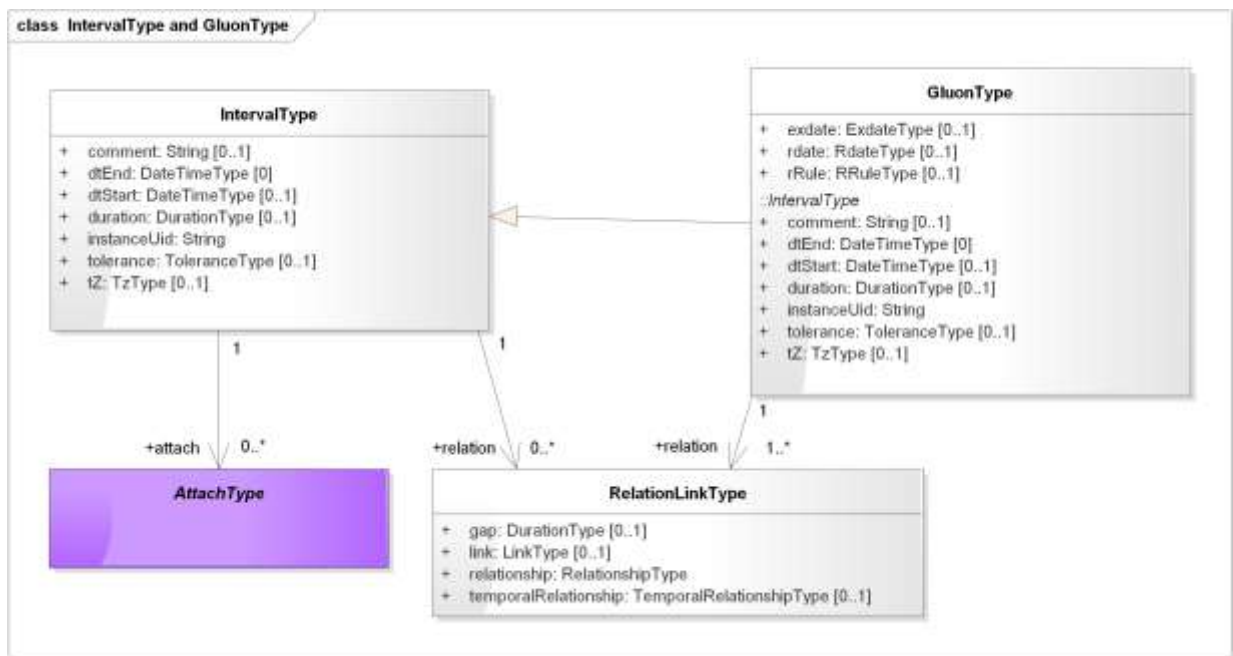
415 The Gluon acts as a reference into a graph of time-related Intervals or Gluons, allowing differing schedule  
416 views depending on the referenced Interval. For example, a room schedule that includes room  
417 preparation, meetings, and room cleanup could have a gluon pointing to the preparation Interval for those  
418 interested in the preparation starting point and associated actions, and another Gluon pointing to the start  
419 of the meetings.

420 *GluonType* has optional recurrence attributes. Recurrence Rules, Exception Dates, and Recurrence  
421 Dates may be necessary to express recurrences, hence are optional. See also Section 4.8.

- 422 • *RRuleType* is an xCal recurrence rule as defined in **[RFC5545]** Section 3.8.5. The recurrence  
423 might be (e.g.) Yearly. The expression in iCalendar syntax is a conformed string.
- 424 • *ExdateType* expresses exception dates **[RFC5545]** Section 3.8.5.
- 425 • *RdateType* expresses recurrence dates **[RFC5545]** Section 3.8.5

426 *GluonType* is a subclass of *IntervalType* with optional recurrence information, and the added requirement  
427 that at least one *RelationLinkType* is associated with a *Gluon*; *IntervalType* has zero or more associated  
428 *RelationLinkType*.

429 **4.5.1 Model Diagram**



430  
431 *Figure 4-5 Gluons, Intervals, and Relationship Links*

432 Note in Figure 4-5 that the minimum number of relations for a Gluon is 1; Intervals need have no  
433 relationships. Only Gluons may have an associated Vavailability.

434 **4.5.2 Discussion**

435 Gluons are Intervals with at least one relation required. One could think of the Gluon as an optional  
436 container for values to “fill in” Interval attributes dynamically and depending on the relationships among  
437 the instances.

438 Note: This technique is used in **[EMIX]** and **[EnergyInterop-v1.0]** to build energy schedules with varying  
439 values but consistent lengths.

440 More generally, a Gluon can be thought of as a pointer into a Sequence, which is a time-related set of  
441 intervals. With Gluons and inheritance rules, missing scheduling information can be dynamically included  
442 in a Sequence.

443 If one considers the unscheduled Sequence and referencing Gluons as a subroutine or template, than a  
444 Gluon defines an instance or invocation of that template.

445 Several Gluons MAY exist (and be advertised) pointing into a given Sequence. When used in this  
446 manner, effectively each Gluon acts as a Service Entry Point for interacting with that template

447 Each Gluon (service entry point) may in turn be associated with additional information: a different price, a  
448 different schedule of availability, and so on. Alternately, a Gluon makes the entire instance associated  
449 with each entry point actionable by scheduling that Sequence. (SeeTable 3-1).

450 If a Gluon includes Recurrence, that Recurrence is not inherited by the referenced Sequence. Rather the  
451 sequence is invoked multiple times in accord with the rules of Recurrence. As an example, consider a  
452 Sequence that lacks only a Start DateTime (*dtStart*) to be scheduled. Recurrence in the Gluon would  
453 define an array of Start DateTimes. The result can be computed by Scheduling that Sequence N times,  
454 once with each element in that array.

455 Recurrence in Gluons MAY NOT be inherited; the recurrence closest to a referenced Sequence is applied  
456 as above.



457 PSMs that claim conformance to this PIM MAY permit inheritance rules for recurrence, in which case they  
458 MUST specify those rules in their conformance statement.

### 459 4.5.3 Relationship to other PIM Components

460 Gluons contain values that may be inherited or overridden in its children in accordance with Section 5.

### 461 4.6 Relationships among Gluons and Intervals

462 Relationships between objects of *IntervalType* are accomplished with *RelationLinkType*. It contains an  
463 abstract class *LinkType* that is a String.

464 The Temporal Relationship and gap together determine the relationship of the referencing Interval and  
465 referenced Interval instances.

466 Note: In **[WS-Calendar]**, **[RFC5545]**, and **[xCal]** the LinkType is a UID, a URI **[RFC3986]**, or a reference  
467 string. This supports both distributed schedules and local identifiers that need not be fully qualified as  
468 would be a UID or a URI. In the PIM, we use a string, without defining the precise type or uses of that  
469 reference—that is left to the PSMs.

470 The gap SHALL be described by class *DurationType*, which has a conformed string to the pattern of  
471 duration **[ISO8601]** extended by **[RFC5545]** to add a “+” or “-” sign.<sup>8</sup> For example, a gap of P-1H with  
472 Temporal Relationship *startToStart* means that the referenced Interval starts one hour before the  
473 referencing Interval.

474 The absence of a sign in the *duration* String SHALL specify a positive value, that is, be treated as if a “+”  
475 sign was present.

476 The absence of a gap attribute in a PIM object SHALL specify a gap of zero duration. An explicit gap of  
477 zero duration may be expressed as e.g. P0H.

478 The *TemporalRelationshipType* enumeration describes the relationship with respect to the referencing  
479 and referenced Interval:

- 480 • *finishToStart* (the conventional, the referenced interval is after the Finish of the referencing  
481 Interval, with an optional gap)
- 482 • *finishToFinish* (the end of the referencing Interval aligns with the end of the referenced Interval,  
483 with an optional gap)
- 484 • *startToFinish* (the start of the referencing Interval aligns with the end of the referenced Interval,  
485 with an optional gap)
- 486 • *startToStart* (the start of the referencing Interval aligns with the start of the referenced Interval,  
487 with an optional gap).

488 RelationshipType SHALL indicate that the linked Interval is a *child* of the linking object.<sup>9</sup>

489 If Relationship Types beyond *child* are available in a PSM, that PSM SHALL describe any values other  
490 than *child* including syntax and semantics in its conformance statement.

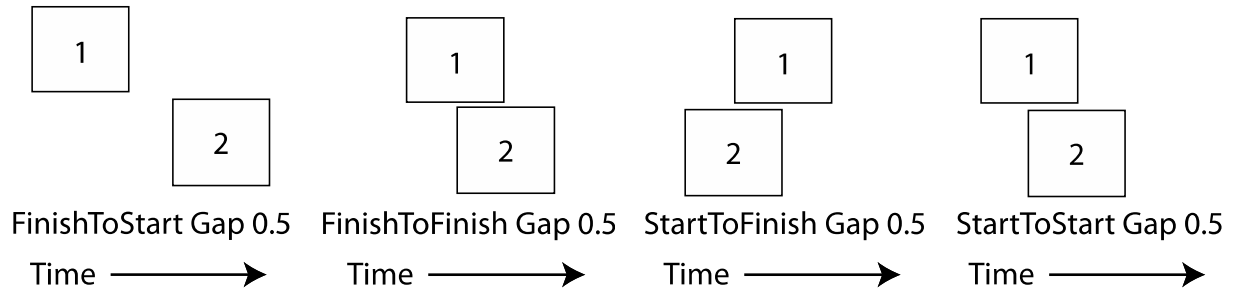
491 Note that the short forms for the temporal relationships listed in **[WS-Calendar]** are not used in the PIM.

492 In Figure 4-6 we show two intervals with each of the temporal relationships. Figure 4-7 shows a gap of  
493 negative 0.5.

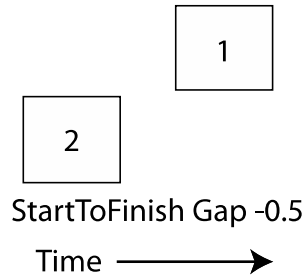
---

<sup>8</sup> **[ISO8601]** duration is unsigned but otherwise more expressive than that in **[RFC5545]** or in **[XMLSchema]**.

<sup>9</sup> **[WS-Calendar]** and **[xCal]**, as do many IETF RFCs, also include in the relationship enumeration an extension point (x-name) and an IANA-registered xCal token (iana-token) **[IANA]**. These are not part of the PIM. **[Relationships]**, an Internet Draft, adds an additional relationship type.

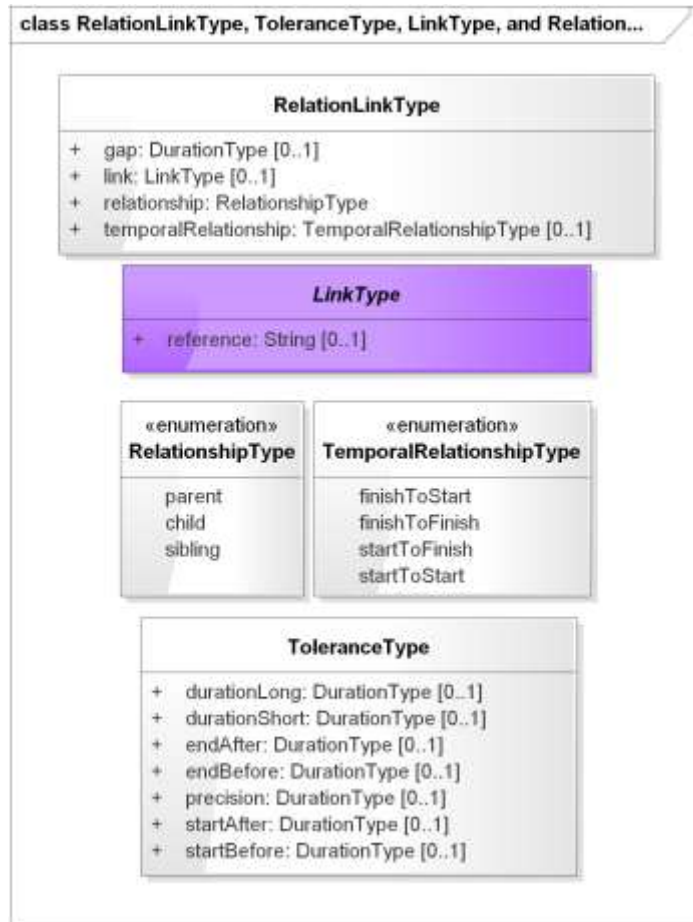


494  
495 *Figure 4-6 Temporal Relationships*



496  
497 *Figure 4-7 Temporal Relationship--startToFinish Negative 0.5 Gap*

498 **4.6.1 Model Diagram**



499  
500 *Figure 4-8 RelationLinkType and Relationship Types*

## 501 4.6.2 Discussion

502 The PIM supports a complete set of the common relationships between time intervals, as used by and  
503 expressed in facility, energy, and other schedules, project management tools, and business process  
504 definitions extending e.g. [BPEL] and [BPMN].

505 The relationships are expressed using the (unsigned) temporal relationship, the (signed) temporal gap  
506 between intervals, and the *RelationshipType* between Gluons and Intervals.

507 A gap SHALL be treated as a signed Duration.

508 ToleranceType attributes SHALL be treated as an unsigned Duration. If a minus sign is present in a  
509 Duration expressing a Tolerance it SHALL be ignored.

510 The PIM SHALL permit only CHILD as *RelationshipType*. Note that other values in the *RelationshipType*  
511 enumeration match those in [RFC6321].

512 Complex structures can be built from primitive relationships, used in data structures, or passed in service  
513 invocations, and interpreted unambiguously.

514 Note: In contrast with the WS-Calendar PSM, *LinkType* contains only a string. The broader range of links  
515 in the WS-Calendar PSM includes a UID, a URI, or other kind of reference (implementation-defined).  
516 Since the abstract link is conceptually a pointer in the PIM, we define a single kind of reference there. It is  
517 maintained as a class to allow a diversity of PSM definitions including but not limited to [WS-Calendar].

518 A PSM claiming conformance to the PIM SHALL document how it defines, manages, and maintains links.  
519 The conformance statement for a PSM SHALL describe uniqueness of references in that PSM.

## 520 4.6.3 Relationship to other PIM Components

521 The PIM allows the common and complete set of temporal relationships between time intervals to be  
522 expressed with optional offsets (the optional *Gap*), while abstracting the details of the relationship into the  
523 *RelationLinkType* class.

524 The abstraction maps cleanly to (e.g.) project management schedules and business process descriptions.

## 525 4.7 Recurrence and the PIM

526 Recurrence (see [RFC5545]) applies both in Availability and in Gluons. See the respective sections for  
527 details.

## 528 4.8 Availability

529 Availability is a means for describing when an actor can be available, or its complement, not available.

530 The WS-Calendar PIM includes the necessary classes to express Availability as in [Vavailability].

531 Note: Historically in iCalendar ([RFC5545] and predecessors), FreeBusy values conveyed information  
532 that is more effectively conveyed by Vavailability. FreeBusy requested all information from a calendar;  
533 Vavailability conveys information for a specific purpose known to the responder.

534 The class *VavailabilityType* includes an interval, which may be partially specified or unspecified, in which  
535 all blocks of *granularity* size are busy per the *busy* attribute – *busy*, *busy-unavailable*, *busy-tentative*.

536 The entire *timeRange* is *busy* for the purposes of a specific use.

537 The class *AvailabilityTimeRangeType* MAY have a start time (optional), and if a start time is present MAY  
538 contain a duration (optional). It can accordingly apply to

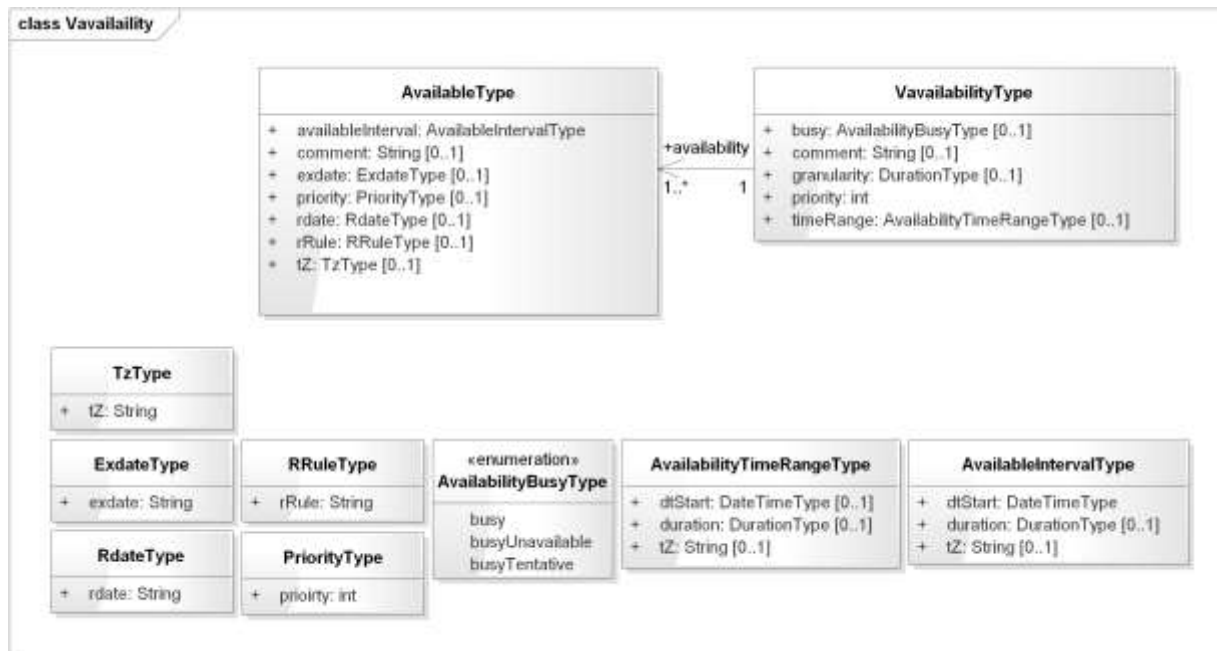
- 539 (1) All time (no dtStart, no duration) (if *timeRange* is not present, all time is indicated)
- 540 (2) A half-infinite interval (dtStart, no duration)
- 541 (3) A bound interval (dtStart, duration)

542 The *granularity* MAY be present and describes the size of the time blocks used for expressing  
543 Availability—for example, for one-hour blocks, *granularity* would be the string P1H.

544 An optional comment is in *VavailabilityType* and in *AvailableType*.

545 Against this backdrop, the associated *AvailableType* objects indicate available times with an  
 546 *AvailableIntervalType* having *dtStart* and optionally *duration* and *tZ* (time zone).  
 547 Note: *AvailableIntervalType* describes a fully bound interval, while *AvailabilityTimeRangeType* describes  
 548 an interval that may be partially or not bound.

### 549 4.8.1 Model Diagram



550  
 551 Figure 4-9 Vavailability and Availability Recurrence Rules

### 552 4.8.2 Discussion

553 The purpose of the Vavailability classes shown in Figure 4-9 is to express the key platform-independent  
 554 semantics for availability. This functionality is an Internet Draft and will be reissued as a Standards Track  
 555 RFC in early 2015.

556 The following recurrence types are drawn from [RFC5545]:

- 557 • *RRuleType* is an xCal recurrence rule as defined in [RFC5545] Section 3.8.5. The recurrence  
 558 might be (e.g.) Yearly. The expression is in iCalendar syntax is a conformed string.
- 559 • *ExdateType* expresses exception dates [RFC5545] Section 3.8.5.
- 560 • *RdateType* expresses recurrence dates [RFC5545] Section 3.8.5
- 561 • *PriorityType* expresses priority for the enclosing AvailableType object [Vavailability] and  
 562 [RFC5545] Section 3.8.1.9.
- 563 • *Comment* is an optional string.

### 564 4.8.3 Relationship to other PIM Components

565 The Availability Package uses recurrence relationships from [Vavailability] [RFC5545] [RFC6321]. This  
 566 allows consistent expression to express availability for (e.g.) Demand Response events in  
 567 [EnergyInterop-v1.0]. Vavailability is not used by other parts of the PIM.

---

## 578 5 Rules for WS-Calendar PIM and Referencing 579 Specifications

570 There are five kinds of conformance that must be addressed for WS-Calendar PIM and for specifications  
571 that claim conformance to this PIM.

- 572 • Conformance to the **inheritance rules**, including the direction of inheritance
- 573 • **Specific attributes** for each type that MUST or MUST NOT be inherited
- 574 • **Conformance rules** that Referencing Specifications MUST follow
- 575 • Description of **Covarying attributes** with respect to the Reference Specification
- 576 • **Semantic Conformance** for the information within the artifacts exchanged

577 We address each of these in the following sections

### 578 5.1 Inheritance in WS-Calendar PIM

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

580 **I1: Proximity Rule** Within a given lineage, inheritance is evaluated though each Parent to the Child  
581 before what the Child bequeaths is evaluated.

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

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

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

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

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

### 596 5.2 Covarying Elements

597 Some attributes of PIM objects may be **covarying**, meaning that they change together. Such elements  
598 are treated as a single element for inheritance: they are either inherited together or the child keeps its  
599 current values intact.

600 Note: This becomes important if one or more of a covarying set have default values.

601 If any covarying attributes are present, then inheritance SHOULD deem they are all present, and  
602 SHOULD assign those without specific definition appropriate default values.

603 A PSM SHALL describe definition and treatment of covarying elements in its conformance statement.

### 604 5.3 Specific Attribute Inheritance

605 In PIM classes the following attributes MUST be inherited in conformance to the Rules (same for Gluons  
606 and Intervals):

- 607 • dtStart
- 608 • dtEnd
- 609 • Duration

- 610       • Designated Interval (Gluon, special upward inheritance rule)
- 611       • Tolerance
- 612    The following attributes MUST NOT be inherited
- 613       • instanceUid (Gluons and Intervals)
- 614       • Temporal Relationships (between Intervals)
- 615       • Relationship Links

---

## 616 6 Conformance

617 This section specifies conformance related to the information model contained in this specification.

### 618 6.1 Conformance for Specifications Claiming Conformance to WS- 619 Calendar PIM

620 Specifications that claim conformance to the WS-Calendar PIM SHALL specify inheritance rules for use  
621 within their specification.

622 These rules SHALL NOT modify the Proximity, Direction, or Override Rules. If the specification includes  
623 covarying attributes, those attributes and their default values SHALL be clearly designated in the  
624 specification and in the PSM conformance statement.

### 625 6.2 General Conformance Issues (Non-Normative)

626 Standards that claim conformance to this specification may need to restrict the variability inherent in the  
627 expressions of Date and Time to improve interoperation within their own interactions. Aspects of Date and  
628 Time that may reward attention and conformance statements include:

- 629 • **Precision** – Does the conforming specification express time in Hours or in milliseconds? Five-  
630 minute intervals? A PSM claiming conformance to this PIM SHOULD select a consistent  
631 precision.
- 632 • **Time Zones and UTC** – Business interactions have a “natural” choice of local, time zone, or UTC  
633 based expression of time. Intents may be local, as they tie to the business processes that drive  
634 them. Tenders may be Time zone based, as they are driven by the local business process, but  
635 may require future action across changes in time and in time zone. Transaction recording may  
636 demand UTC, for complete unambiguity. The specification cannot require one or another, but  
637 particular business processes may require appropriate conformance statements. A PSM claiming  
638 conformance to this PIM SHALL detail Time Zone treatment as well as assumptions and implicit  
639 values.
- 640 • **Business Purpose** – The PIM does not distinguish between different uses of objects that may  
641 have different purposes. For example, a general indication of capability and/or timeliness may be  
642 appropriate for a market tender, and an unanchored Sequence may be appropriate. In the same  
643 specification, performance execution could require merely that the Gluon Anchor the Interval. If  
644 the distinction between Unanchored and Anchored Interval is necessary for a particular use, the  
645 PSM claiming conformance SHALL indicate the proper form for each of its uses.

### 646 6.3 Conformance of Intervals

#### 647 6.3.1 Intervals and Gluons

648 Intervals SHALL have *duration* AND optionally *dtStart*. If a non-compliant Interval is received in a service  
649 operation or by reference with a value for *dtEnd*, then *dtEnd* SHALL be ignored.

650 Within a Sequence, at most one Interval MAY have a *dtStart* or a *dtEnd*.

651 Specifications that claim conformance SHALL define the business meaning of zero duration Intervals or  
652 prohibit zero duration intervals, and include that definition or prohibition in their conformance statement.

#### 653 6.3.2 Other Attributes

654 A Gluon MAY have a *dtStart* value.

## 655 **6.4 Conformance of Bound Intervals and Sequences**

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

- 659 • Intervals SHALL have values assigned for dtStart and duration, either explicitly or through  
660 inheritance
- 661 • Intervals SHALL have no value assigned for dtEnd
- 662 • Within a Sequence at most the Designated Interval may have dtStart and duration with a value  
663 specified or inherited.
- 664 • If Sequences are composed to create other Sequences, then the Designated Intervals within the  
665 composing Sequence are ignored.
- 666 • Any specification claiming conformance to the WS-Calendar PIM MUST satisfy all of the following  
667 conditions:
  - 668 ○ Follow the same style of inheritance (per the Rules)
  - 669 ○ Specify attribute inheritability in the specification claiming conformance
  - 670 ○ Specify whether certain sets of elements must be inherited as a group or specify that all  
671 elements can be inherited or not on an individual basis

## 672 **6.5 Security Considerations (Non-Normative)**

673 The WS-Calendar PIM describes an informational model. Specifications claiming conformance with the  
674 WS-Calendar PIM are likely to use the schedule and interval information as but a small part of their  
675 overall communications.

676 Specifications involving communication and messages that claim conformance to this specification should  
677 select the communication and select from well-known methods to secure that communication appropriate  
678 to the information exchanged, while paying heed to the costs of both communication failure and of  
679 inappropriate disclosure. To the extent that iCalendar schedule servers are used, the capabilities of  
680 security of those systems should be considered as well. Those concerns are out of scope for this  
681 specification.



## 682 7 Examples using the PIM (Non-Normative)

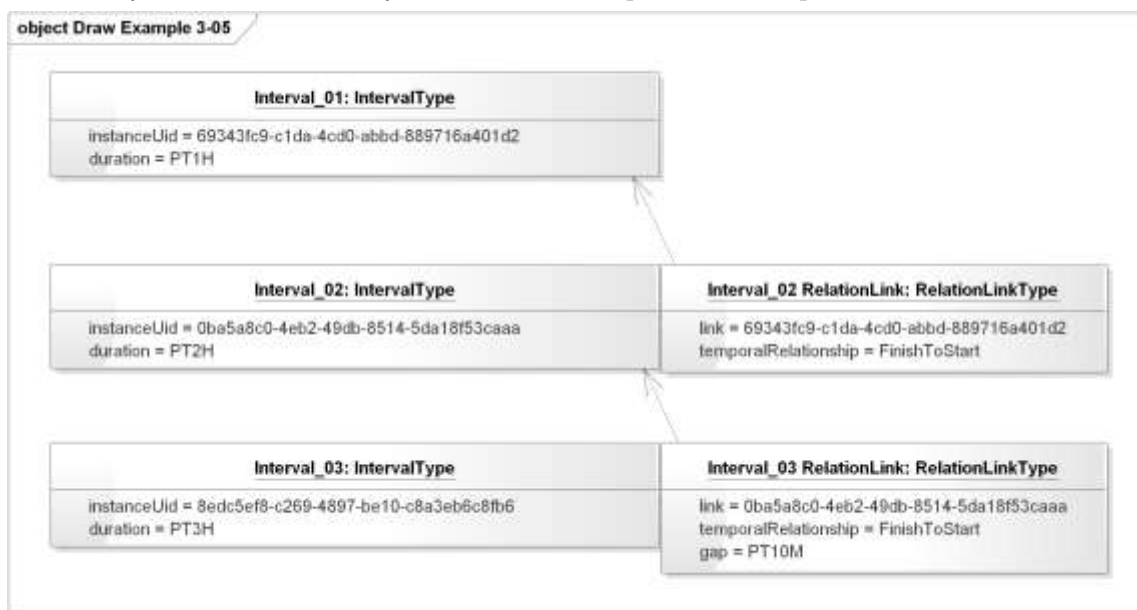
683 We include several examples drawn from a variety of sources. These examples were created to illustrate  
684 facility scheduling, energy scheduling, and related topics.

685 The dashed lines in the Object Diagrams are not UML, but are a graphical depiction of the links, with the  
686 head of the arrow indicating the referenced (linked) Interval and the tail indicating the referencing (linking)

687 A separate Committee Technical Note **[PIM Examples]** is in progress with examples including ones  
688 drawn from those in **[WS-Calendar]** and other specifications.

### 689 7.1 Related Intervals

690 This example is based on Example 3-05, line 483 in **[WS-Calendar]**.



691  
692 Figure 7-1 PIM Expression of WS-Calendar Examples 3-05

693 In this diagram, a Gluon could refer to the Sequence with a reference to Interval\_03.

### 694 7.2 A Meeting Schedule

695 Consider a meeting scheduled for a specific time – say 2pm and lasting two hours.

696 The meeting itself can be represented (and scheduled with attendees) as a single interval with duration 2  
697 hours.

698 To carry out the meeting, there are other activities both before and after, and possibly during, the meeting  
699 time. See Figure 7-2 Simple Meeting Schedule below.

700 First, the room needs to be set up for the meeting. The Heating, Ventilating, and Air Conditioning system  
701 (HVAC) may need to pre-cool the room for the scheduled number of attendees. And the room needs to  
702 be cleaned up before setup for the next meeting.

703 Each of these activities can be scheduled separately, and done by different actors. But they need to be  
704 completed to set up and restore the room.

705 Also consider a pre-meeting of the leaders in the room, starting 30 minutes before the main meeting, and  
706 lasting 20 minutes so the leaders can meet and greet attendees.

707 The gluons on the right are references into the sequence of intervals; the respective sequences are *child*  
708 to the respective Gluons.

709 (1) The start of the HVAC pre-cooling is given to the HVAC control system  
710 (2) The start of the main meeting gluon is given to the meeting attendees

711 Additional gluons could be given to (e.g.) the room set-up team, pointing to the Prepare Room interval,  
712 and to the Pre-Meeting interval for the meeting leaders.

713 Additional elaboration might include the pre-purchase of energy for the pre-cooling (or committing in an  
714 energy schedule, which the HVAC control system uses to balance energy use through the day to avoid  
715 demand charges.

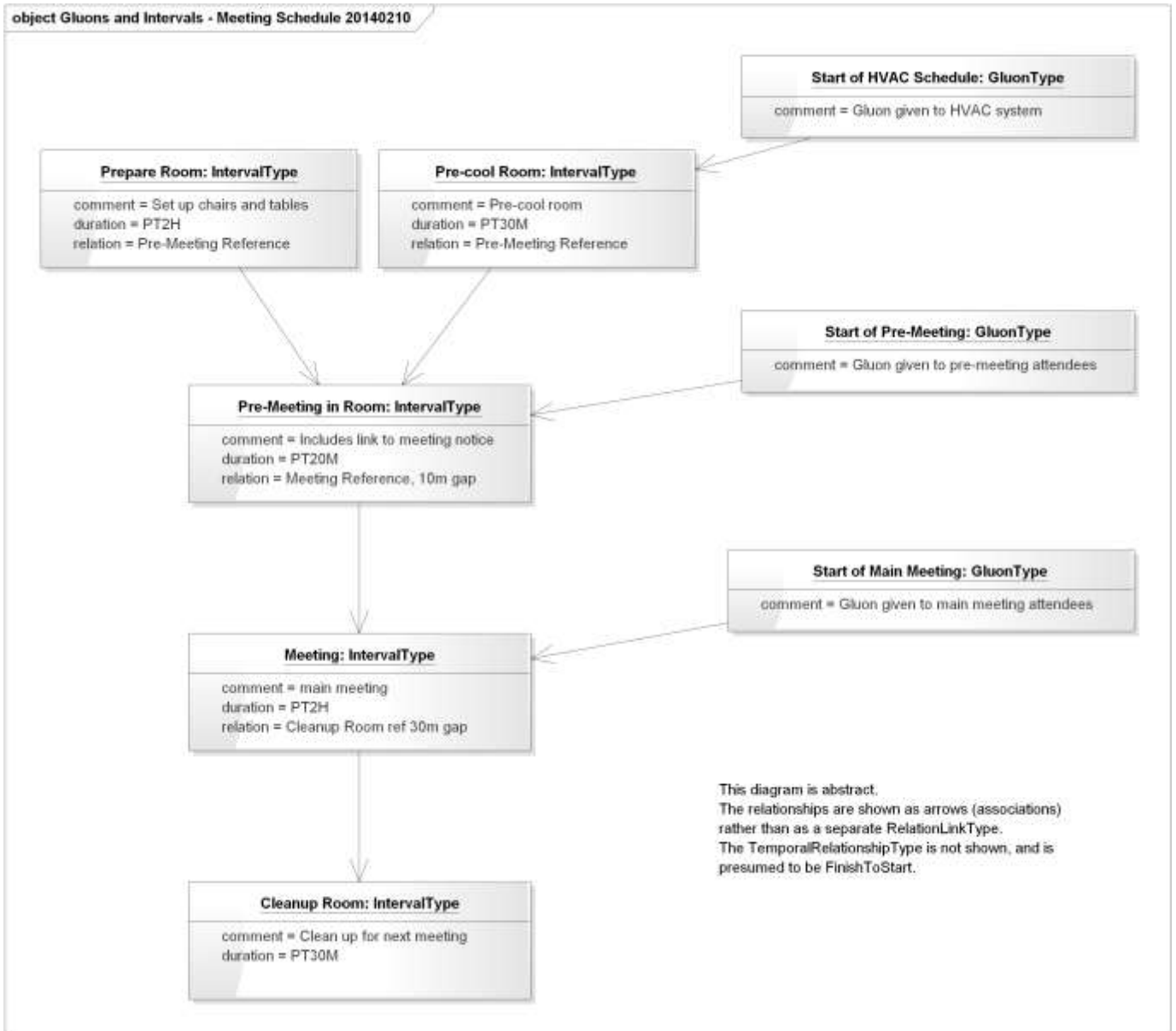
716 Finally, the actions are all based on where you reference the schedule—working back from the start time  
717 (inherited from the start of main meeting gluon) the pre-meeting is 30 minutes earlier, and the setup is 2  
718 hours and 30 minutes earlier.

719 The HVAC schedule gluon might be all that the control system needs, combined with the knowledge from  
720 the schedule that the meeting is over in 2 hours 30 minutes after the 30-minute pre-cool period, and that  
721 cleanup takes another 30 minutes.

722 We have not tried to show all possible schedules and variations – perhaps the setup takes longer but is  
723 finished earlier, using an *endBefore* tolerance (and a zero *endAfter* tolerance).

724 Note that this schedule may be used for any meeting – the start time can be placed in a gluon that  
725 references the Meeting interval. Likewise, the length could also be inherited from that same gluon. The  
726 structure of the schedule would be determined by facility policy (e.g. “you must allow two hours for  
727 setup”), and the schedule itself is relocatable and reusable.

728 The figure is informal, and does not reflect all the details of relationships (the arrows indicate the  
729 relationships which are not otherwise shown with relations and IDs).



730

731 *Figure 7-2 Simple Meeting Schedule*

732

---

## Appendix A. Acknowledgments

733

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

734

735

### Participants:

Bruce Bartell	Southern California Edison
Chris Bogen	US Department of Defense (DoD)
Edward Cazalet	Individual
Toby Considine	University of North Carolina at Chapel Hill
Robin Cover	OASIS
William Cox	Individual
Sharon Dinges	Trane
Michael Douglass	Rensselaer Polytechnic Institute
Craig Gemmill	Tridium, Inc.
Dave Hardin	EnerNOC
Gale Horst	Electric Power Research Institute (EPRI)
Gershon Janssen	Individual
Ed Koch	Akuacom Inc.
Benoit Lepeuple	LonMark International
Carl Mattocks	Individual
Robert Old	Siemens AG
Joshua Phillips	ISO/RTO Council (IRC)
Jeremy Roberts	LonMark International
David Thewlis	CalConnect

736

## Appendix B. Revision History

Revision	Date	Editor	Changes Made
01	November 15 2012	William Cox	Initial Draft based on contributed models
02	December 20 2012	William Cox	First draft conformance section. Added explanatory text in individual model sections. GluonType is now a subclass of IntervalType, rather than GluonType having an association to IntervalType.
03	January 31, 2012	William Cox	Completed most sections; indicated questions for the TC as "EDITOR'S NOTE"s. Model is the same as for WD02. WD03 contains a quotation with modifications from the WS-Calendar conformance sections.
04	April 10, 2013	William Cox	Update with responses to questions from WD03; minor changes to the model and many clarifications based on meeting discussions. Included differences between the normative semantics and conformance sections and WS-Calendar 1.0 as non-normative Appendices.
05	April 24, 2013	William Cox	Addressed remaining Editor's Notes from previous Working Drafts. Changed cardinality for attachment from [1..1] to [0..1] in parallel with unbound attributes expressed in UML. Prepared text for public review.
06	16 January 2014	William Cox	Simplification of relations and LinkType. Addition of instance (object) diagrams to express examples. Includes PIM to WS-Calendar-as-PSM mapping.
07	17 January 2014	William Cox	Addresses comments from TC review of WD06. Eliminated unused DurationParameterEnum, corrected gap to DurationStringType (with no tolerance values), eliminated iana-token and x-name relationship types. Identified but did not correct the application of tolerance to dtStart, dtEnd, and duration. Clarified intended sources of examples. Eliminated unused classes and objects in the model.
08	13 March 2014	William Cox	Simplifies the DurationType, moves tolerance to IntervalType instead of the former DurationValueType. Completed PIM-PSM mapping, updated references, other editorial and technical clarity change. Updated diagrams to express updated model.

Revision	Date	Editor	Changes Made
09	21 April 2014	William Cox	First inclusion of mapping descriptions. Clarified DateTimeType and DurationType relationship to ISO 8601. Many minor edits; minor model changes.
10	08 May 2014	William Cox	Edits throughout based on meeting discussion. lowerCamelCase for ToleranceType, textual changes, and updated diagrams.
11	31 July 2014	William Cox	Address comments from second Public Review. Normative reference to and comparisons to WS-Calendar have largely been removed. Much text has been moved to non-normative sections or appendices. Diagrams and the model were updated.
12	03 August 2014	William Cox	Completed addressing comments from second Public review. Significant modifications to Availability, and simplification of DurationType. Deleted FreeBusy. Detailed corrections to attribute names to align with model. Model updated to reflect corrections, and all figures for PIM UML were updated.
13 and Committee Specification 01	14 August 2014	William Cox	Address one comment from TC members, changed all attributes including those in enumerations to lowerCamelCase (ToleranceType, BusyType, RelationshipType). Minor editorial corrections. Use of italic is more consistent except in Appendix C.
14	13 March 2015	William Cox	Addresses many comments subsequent to CS01/WD13. Changes include cardinality refinements, integration of Vavailability in the PIM, Recurrence classes including Rrules, Exdate, simplification of type naming, and NIEM-like separation of conformed string definitions from classes. DateType and TimeType were included (DateType from ISO8601, TimeType after XML Schema).
15	22 April 2015	William Cox	Changes to address Public Review and other comments. Corrected Vavailability classes, added recurrences to Gluons consistent with RFC5545. Editorial corrections. Defined recurrence inheritance as disallowed for Gluons.

---

## 740 Appendix C. PIM to WS-Calendar PSM Transformation

741 MDA instances include a Platform-Independent Model (PIM), defined in this specification, and a  
742 transformation to one or more Platform-Dependent Model (PSM). In this section we briefly describe the  
743 mapping from this PIM to **[WS-Calendar]** (considered as a PSM).

744 Largely the same data types and conformed strings for instance values are used in the PIM, to ensure  
745 that the transformation is straightforward.

746 Diagrams with golden class backgrounds are from **[WS-Calendar]**; diagrams with light class backgrounds  
747 are from this PIM.

748 A UML model for WS-Calendar is of a different style from this PIM. WS-Calendar expresses the  
749 information for Intervals, Gluons, and other classes in terms of collections of Parameters, Properties, and  
750 Value Types, held in those collections with others that may not reflect the abstractions of WS-Calendar.

### 751 C.1 General Transformations

752 On inspection the transformations between the PIM model and the **[XMLSchema]** for **[WS-Calendar]** are  
753 generally clear. The classes in the PIM are similar or identical to those in **[WS-Calendar]** including  
754 attribute/element names, but are arranged as simple classes rather than collections of properties within a  
755 potentially larger set of properties.

### 756 C.2 Specific Transformations

757 In the following subsections we describe transformations from the PIM to the WS-Calendar PSM.

758 In WS-Calendar an Interval or Gluon is a Vcalendar component, expressed as a subclass of  
759 ICalendar::VcalendarContainedComponentType.

760 That class informally contains sets of Properties, Values, and Parameters, based on the widely used  
761 iCalendar definition. The PIM does not distinguish between parameters, values, and properties and the  
762 differing types.

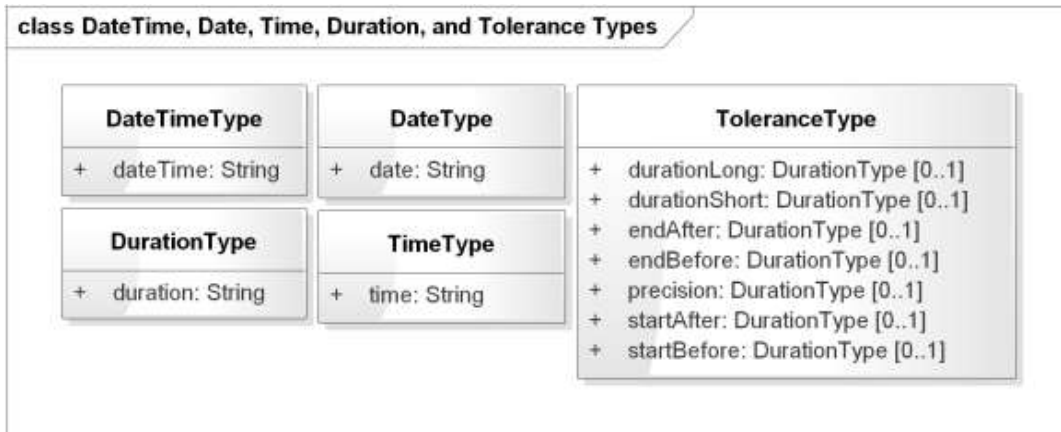
763 In the subsections below we describe the transformations for

- 764 • DateTime and Duration Types, the fundamental types for talking about time and schedule
- 765 • ToleranceType
- 766 • Intervals and Gluons
- 767 • Relationships
- 768 • Vavailability

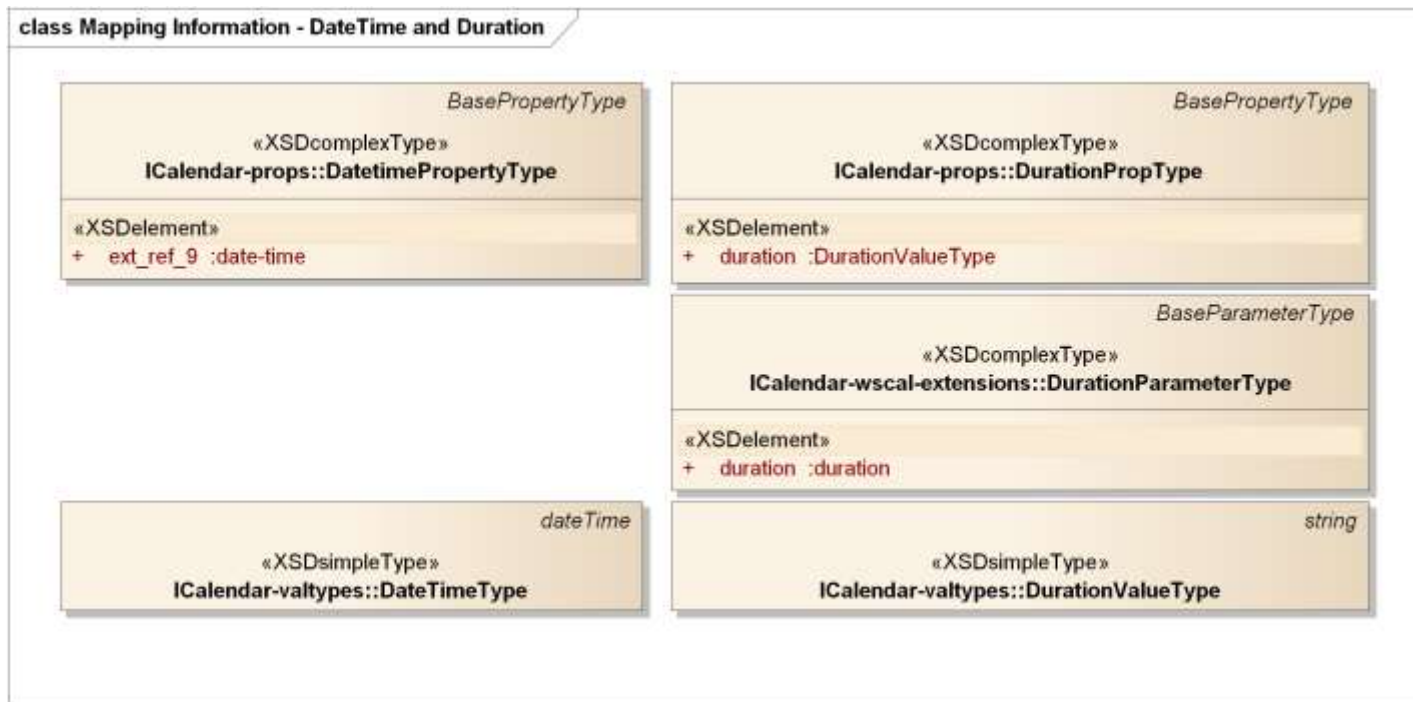
#### 769 C.2.1 Transformation for DateTime and Duration Types

770 DateTimeType and DurationType use **[ISO8601]** conformed strings. In transforming objects of these PIM  
771 classes the values must be expressible in the target PSM. The following two figures show selected WS-  
772 Calendar classes and the PIM classes DateTimeType, DurationType, and ToleranceType.

773 There are different conformed strings for DurationType and DateTimeType. The PIM uses **[ISO8601]**  
774 duration and date time semantics; these are isolated in the PIM classes DateTimeType and DurationType  
775 to facilitate mapping to classes in PSMs including those based on **[WS-Calendar]** and **[XMLSchema]**.



776  
777 *Figure 7-3 PIM Source Classes for DateTimeType and Duration Types*  
778

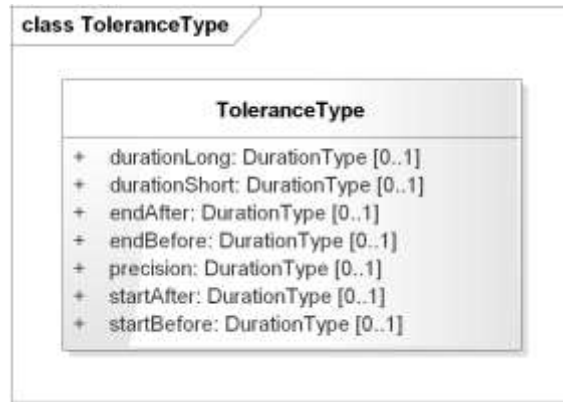


779  
780 *Figure 7-4 WS-Calendar Target Classes*  
781  
782 *Table 7-1 PIM to PSM Mapping for DateTimeType and Duration Types*

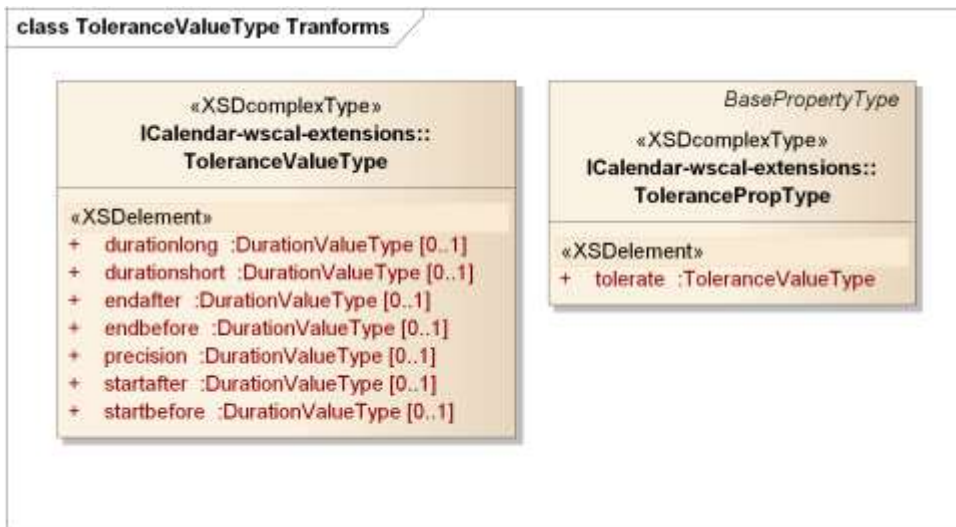
PIM Class Name	WS-Calendar Class Name	Notes
DateTimeType	ICalendar-valtypes::DateTimeType	Restrictions on ISO8601 strings when mapped to RFC5545 strings
DurationType	ICalendar-valtypes::DurationValueType	Restrictions on ISO8601 duration strings when mapped to RFC5545 strings; Gap is signed as in <b>[WS-Calendar]</b> . DurationType must be non-negative for ToleranceType.



783 **C.2.2 Transformation for Tolerance Type**



784  
785 *Figure 7-5 PIM Source Class for ToleranceType*



786  
787 *Figure 7-6 WS-Calendar Target Classes for Tolerance Type*

788 The PIM ToleranceType is identical with minor differences in attribute names and types to the WS-  
789 Calendar class with the same function, as shown in Figure 7-5 and Figure 7-6 above.

790 The differences are

- 791 • The PIM uses *DurationType* rather than the WS-Calendar *DurationValueType*
- 792 • The PIM uses *ToleranceType* rather than the WS-Calendar *ToleranceValueType*
- 793 • The PIM attribute names are in lowerCamelCase rather than lower case.

794

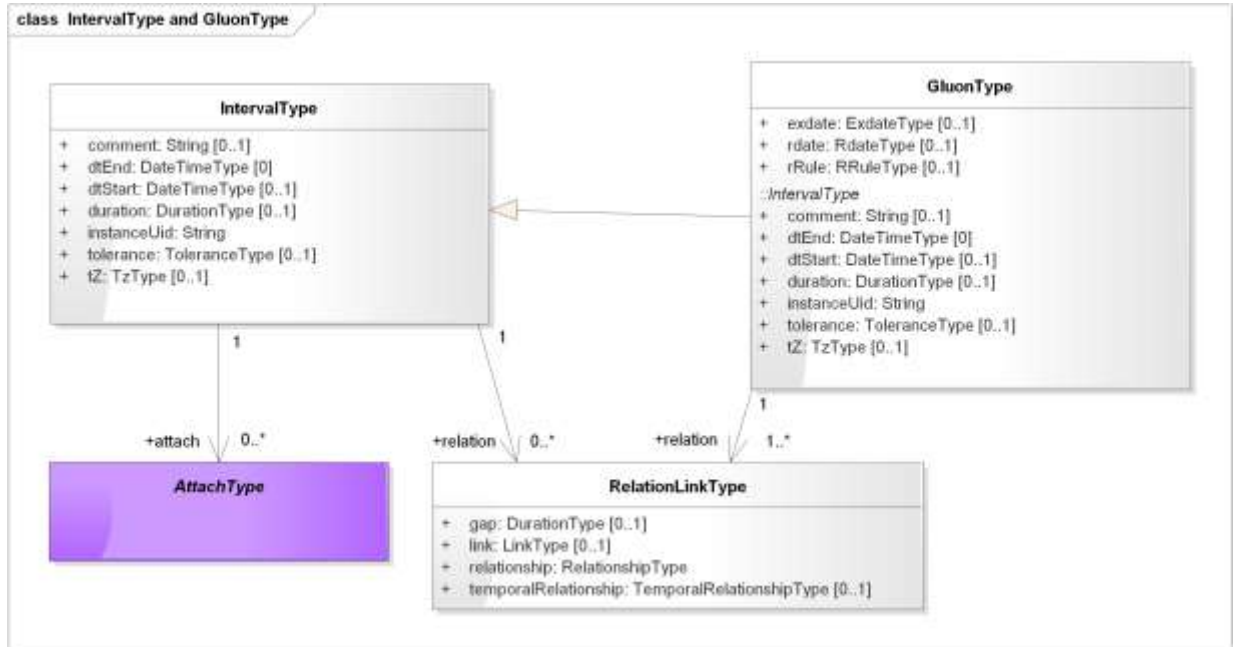
795 *Table 7-2 PIM to PSM Mapping for ToleranceType*

PIM Class Name	WS-Calendar Class Name	Notes
ToleranceType	ICalendar-wscal-extensions::ToleranceValueType	Attributes map respectively to attributes of the same name with lowerCamelCase in PIM <i>ToleranceType</i> . Types map per Section C.2.1.

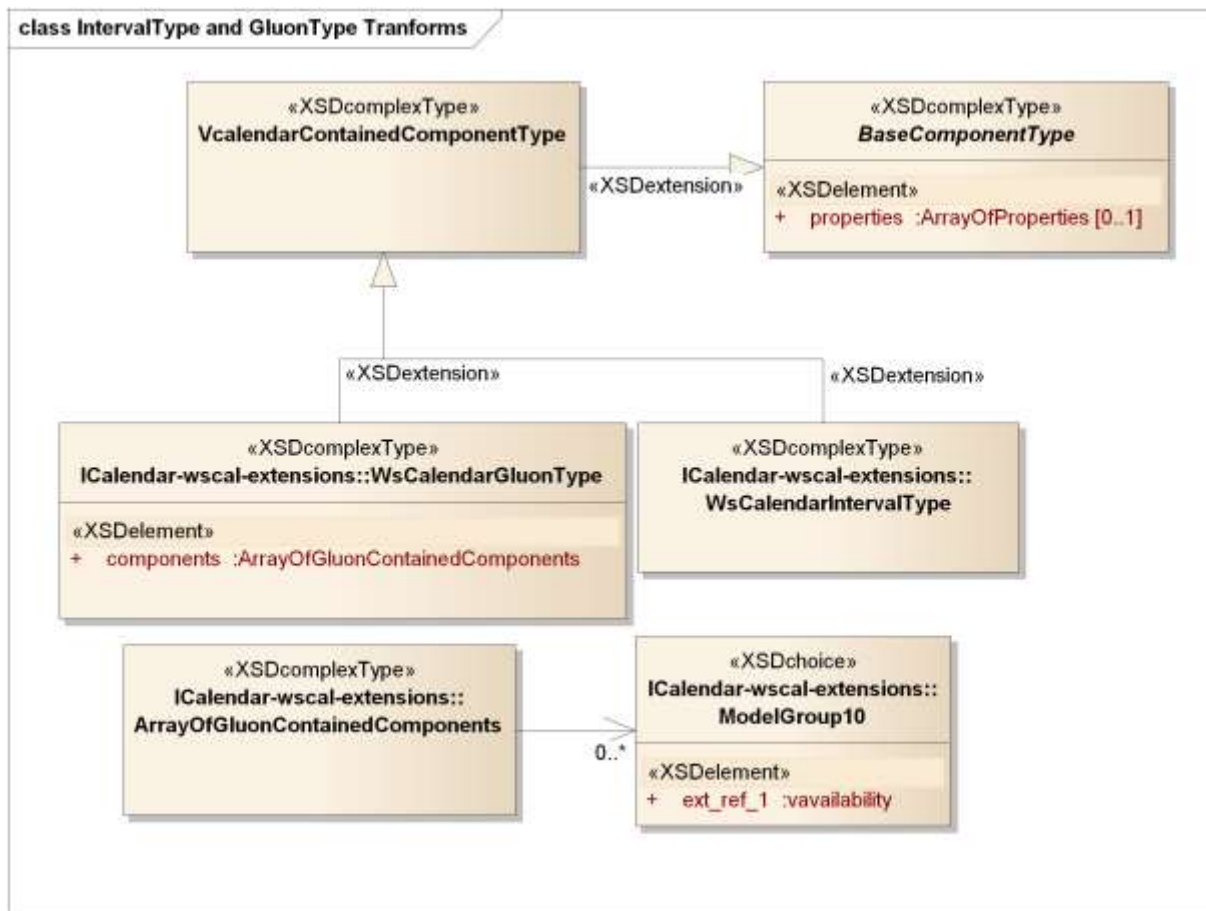
796 **C.2.3 Transformation for Interval and Gluon Types**

797 We treat the Gluon and Interval together; GluonType is a subclass of IntervalType, and extends  
 798 IntervalType as shown in Figure 7-7:

- 799 • Changing the cardinality of the attribute *relation* to require one or more *relations*  
 800 • Optionally including *Vavailability*



801  
 802 *Figure 7-7 PIM IntervalType and GluonType*



803

804 *Figure 7-8 WS-Calendar Target IntervalType and GluonType*

805 A WS-Calendar Interval (and its subclass Gluon) is a Vcalendar object, with a set of properties, values,  
 806 and parameters optionally included. Among those are the attributes of the PIM IntervalType, essentially  
 807 the same set of attributes of GluonType, and the additional VavailabilityType in GluonType.

808 Properties with the same semantics and value types exist in WS-Calendar as well as the PIM; the name  
 809 and type transformations are described in the following table. RelationLinkType is addressed in the next  
 810 section.

811 *Table 7-3 PIM to PSM Mapping for IntervalType and GluonType*

PIM Attribute and Type	WS-Calendar Target Type	Notes
comment: string	ICalendar-Props::CommentPropType	Target takes a text value.
dtEnd: DateTimeType	ICalendar-Props::DtendPropType	Constrained string per <b>[RFC5545]</b>
dtStart: DateTimeType	ICalendar-Props::DtstartPropType	Constrained string per <b>[RFC5545]</b>
duration: DurationType	ICalendar-wscal-extensions::DurationPropType	Constrained string per <b>[RFC5545]</b>
instanceUid: string	ICalendar-Props::UidPropType	

PIM Attribute and Type	WS-Calendar Target Type	Notes
tolerance: ToleranceType	ICalendar-wscal-extensions::ToleranceValueType	Attribute of TolerancePropType is <i>tolerate</i>
tZ: string	ICalendar-Props::TzidPropType	Constrained string per <b>[RFC5545]</b>
Relation: RelationLinkType	ICalendar-link-extension::LinkPropType	Target has possible UID, URI, Reference attributes

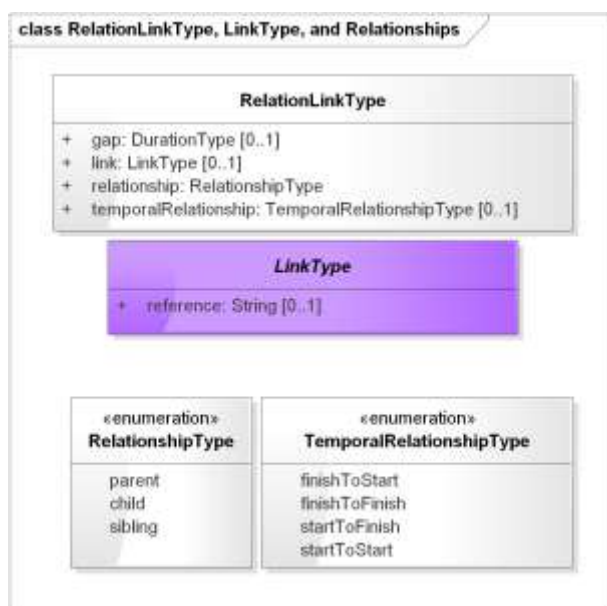
812 For the Recurrence attributes (Gluons only) the transformation is from the attribute value in GluonType to  
813 the appropriate ValueTypes in **[WS-Calendar]**.

## 814 C.2.4 Transformation for Relationships

815 In this section we detail transformations for *RelationLinkType* and for its attributes: *link*, *relationship*, and  
816 *temporalRelationship*.

817 Both **[WS-Calendar]** and the current draft extending iCalendar **[Relationships]** have a single  
818 *ReltypeParamType* which combines relationships (e.g. CHILD) and temporal relationships (e.g.  
819 FinishToStart) in one.

820 In the PIM we maintain separate attributes of RelationLinkType for those two classes of relationship, and  
821 separate enumerations, *RelationshipType* and *TemporalRelationshipType*, rather than multiple parameter  
822 values. This mirrors current programming practices favoring explicit unitary value enumerations rather  
823 than logically combining a set of values. Moreover, the use of text *reltypes* adds brackets around every  
824 string no matter how short. The implicit repetition of a parameter, each with its attendant brackets, may  
825 not be a correct interpretation.



826  
827 *Figure 7-9 PIM RelationLinkType, LinkType, RelationshipType, and TemporalRelationshipType*

828 The following table describes the transformation in detail for the classes and enumerations in Figure 7-9.  
829 The *related-to* property in WS-Calendar may include a *reltype* parameter.

830 The “short form” in Temporal Relationships Table 3-2, line 423 in **[WS-Calendar]** is not used in the PIM;  
831 transformation should be to the “long form” in **[WS-Calendar]**.

832 The values for RelationshipType and TemporalRelationshipType map to the same names in WS-  
833 Calendar, excepting only that TemporalRelationshipType in WS-Calendar is all lower case rather than  
834 lowerCamelCase.

835 Table 7-4 PIM to PSM Mapping for Attributes of PIM RelationshipType and TemporalRelationshipType

PIM Enumeration	WS-Calendar Target Type	Notes
RelationshipTypes:: PARENT, CHILD, SIBLING	ICalendar- props::RelatedToPropType	PIM uses only CHILD
TemporalRelationshipType:: FinishToStart, FinishToFinish, StartToFinish, StartToStart	ICalendar- props::RelatedToPropType	

836

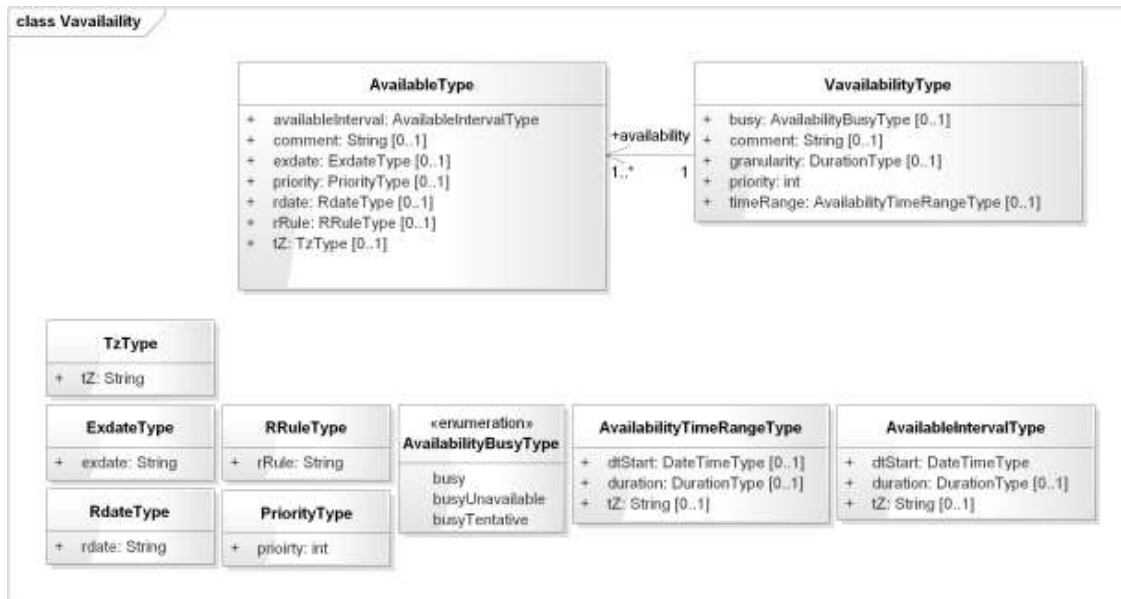
PIM Attribute and Type	WS-Calendar Target Type	Notes
gap: DurationType	ICalendar- Params::DurationParameterType	Duration is an <b>[ISO8601]</b> conformed string that maps to a constrained string per <b>[RFC5545]</b> with optional sign.
Link: LinkType	ICalendar-props::RelatedToPropType	All of the RelatedToPropType extended choices are strings ( <i>uri</i> , <i>uid</i> , and <i>text</i> ). PIM <i>LinkType</i> is an abstract type with a <i>String</i> attribute.
Relationship: RelationshipType	iCalendar-params:: ReltypeParamType – iCalendar- props::related-to: RelatedToPropType	In the same set of RelatedToPropType as temporal relationships
temporalRelationship: TemporalRelationshipType	iCalendar-params:: ReltypeParamType – iCalendar- props::related-to: RelatedToPropType	In the same set of RelatedToPropType as relationships

837 Table 7-5 PIM to PSM Mapping for Enumeration Members

838

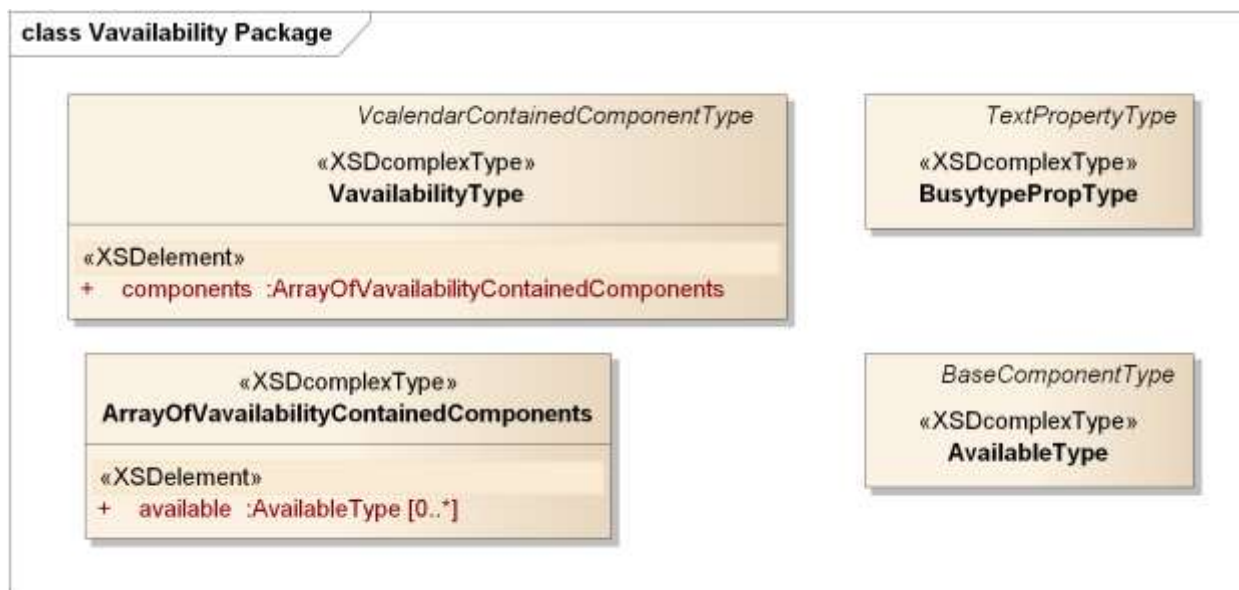
## 839 C.2.5 Transformation for Vavailability and FreeBusy

840 The Vavailability classes will be synchronized with the final Standards Track RFC after **[Vavailability]**  
841 completes the IETF process.



842  
843 Figure 7-10 PIM Vavailability Package Classes

844 The Vavailability package in iCalendar-availability-extension.xsd is as follows:



845  
846 Figure 7-11 Vavailability Package from iCalendar-availability-extension

847 The VavailabilityType has zero or more AvailableType objects inside. The AvailableIntervalType is implicit  
848 in the way components are defined in [RFC5545] and [RFC6321]. The Recurrence attributes map is from  
849 the attribute value in AvailabilityType to the appropriate ValueTypes in [WS-Calendar].

850

---

851 **Appendix D. PIM to IEC TC57 CIM Intervals and**  
852 **Sequences (Non-Normative Example)**

853 The IEC TC57 Common Information Model **[IEC CIM]** uses time intervals in a variety of ways. We  
854 describe straightforward transformations in both directions between

- 855 • A fully bound PIM interval (which uses *dtStart* and duration and time zone) with an *Attach*
- 856 • To a CIM interval (which uses *dtStart* and *dtEnd* in UTC).

857 First we must understand that a time interval *per se* does not exist in **[IEC CIM]**. Instead, explicit  
858 *dtStart* and *dtEnd* attributes are included, often as a timestamp value. In some part of the CIM model the  
859 start and end are implicit. In short in the CIM model is a great variety of expression for time intervals, and  
860 all are expressed by including attributes in a class, not something that has a separate definition.

861 The mapping is from the appropriate calculated or explicit values in an object to an Interval, setting the  
862 *dtStart*, *tZ*, and *duration* in the PIM *IntervalType*.

863 The CIM also assumes UTC, which must be an explicit time zone in iCalendar.

- 864 (1) The CIM class *dtStart* maps to *dtStart* and *tZ* with value UTC in the PIM Interval
- 865 (2) The CIM class *dtEnd* is used with CIM *dtStart* to compute the PIM *duration*
- 866 (3) The CIM class is mapped to an *AttachType* created in a PIM or PSM model

867 The other direction is also straightforward:

- 868 1. Determine the CIM class as target from the concrete *AttachType*
- 869 2. The PIM *dtStart* and *tZ* is mapped to the appropriate UTC time, and placed in the CIM class  
870 *dtStart*
- 871 3. The PIM duration is added to the CIM *dtStart* and placed in CIM *dtEnd*

872 Because of the interval attributes inserted in each data item (or implicitly present), and because CIM  
873 intervals include *dtStart* and *dtEnd*, sequences of CIM intervals are not relocatable in the same way as  
874 PIM and **[WS-Calendar]** Intervals.

875 To relocate, a Sequence of CIM intervals must each be modified with the new *dtStart* and *dtEnd*.

876 In the PIM changing *dtStart* in the Designated Interval or in a referencing Gluon relocates a sequence.

877