CalConnect TC

Calendar and scheduling — iCalendar Transport-Independent Interoperability Protocol (iTIP)

Published Standard

The Calendaring and Scheduling Consortium, Inc. 2024

Contents

Abstract......iv

Introduction......v
Related Documents......v
Roles......v
Methods......v

1. Normative references......1

2. The Abstract Data Model......1
2.1. Calendar Address......1
2.2. The Owner......2
2.3. Application Protocol Elements......2

3. Security Considerations......5

4. IANA Considerations......6
4.1. Initialization of the Status registry......6

5. Acknowledgements......6

Bibliography......7

Abstract

This document specifies a protocol that uses calendar objects to provide scheduling interoperability between different calendaring systems. This is done without reference to a specific transport protocol so as to allow multiple methods of communication between systems. Other documents define profiles of this protocol that use specific, interoperable methods of communication between systems.

The iCalendar Transport-Independent Interoperability Protocol (iTIP) complements the calendar object specifications by adding semantics for group scheduling methods commonly available in current calendaring systems. These scheduling methods permit two or more calendaring systems to perform transactions such as publishing, scheduling, rescheduling, responding to scheduling requests, negotiating changes, or canceling.

Introduction

This document specifies how calendaring systems use calendar objects to interoperate with other calendaring systems. In particular, it specifies how to schedule events and tasks. It further specifies how to search for available busy time information. It does so in a general way, without specifying how communication between different systems actually takes place. Other documents specify transport bindings between systems that use this protocol.

The protocol is described in abstract terms with examples shown in iCalendar as defined in IETF RFC 5545 and jsCalendar as specified in IETF RFC 8984.

This protocol is based on messages sent from an originator to one or more recipients. For certain types of messages, a recipient may reply in order to update their status and may also return transaction/request status information. The protocol supports the ability for the message originator to modify or cancel the original message. The protocol also supports the ability for recipients to suggest changes to the originator of a message. The elements of the protocol also define the user roles for its transactions.

This specification obsoletes RFC 5546.

Related Documents

Implementers will need to be familiar with several other specifications that, along with this one, describe the Internet calendaring and scheduling standards. The related documents are:

IETF RFC 5545	The iCalendqr specification for the objects, data types, properties, and property parameters used in the protocols in that format. Also defines the methods for representing and encoding them.
IETF RFC 8984	Describes the properties and objects used in the jsCalendar representation.
IETF RFC 6047	specifies an Internet email binding for iTIP.

This specification does not attempt to repeat the concepts or definitions from these other specifications. Where possible, explicit references are made to the other specifications.

Roles

Exchanges of calendar objects for the purposes of group calendaring and scheduling occur between “Calendar Users” (CUs). CUs take on several roles in iTIP:

Table 1

Role	**Description**
Organizer	The CU who initiates an exchange takes on the role of Organizer. For example, the CU who proposes a group meeting is the Organizer.
Attendee or participant	CUs who are included in the scheduling message as possible recipients of that scheduling message. For example, the CUs asked to participate in a group meeting by the Organizer take on the role of Attendee.
Other CU	A CU that is not explicitly included in a scheduling message, i.e., not the Organizer or an Attendee.

Note that in iCalendar and jsCalendar the role is also used to convey descriptive context about an “Attendee” — such as “chair”, “required participant”, or “non- required participant”. These roles have nothing to do with the calendaring workflow.

Methods

The iTIP methods are listed below and their usage and semantics are defined in Section ??? of this document.

Table 2

Method	Description
PUBLISH	Used to publish a calendar object to one or more “Calendar Users”. There is no interactivity between the publisher and any other “Calendar User”. An example might include a baseball team publishing its schedule to the public.
REQUEST	Used to schedule a calendar object with other “Calendar Users”. Requests are interactive in that they require the receiver to respond using the reply methods. Meeting requests, busy-time requests, and the assignment of tasks to other “Calendar Users” are all examples. Requests are also used by the Organizer to update the status of a calendar object.
REPLY	A reply is used in response to a request to convey Attendee status to the Organizer. Replies are commonly used to respond to meeting and task requests.
ADD	Add one or more new instances to an existing recurring calendar object.
CANCEL	Cancel one or more instances of an existing calendar object.
REFRESH	Used by an Attendee to request the latest version of a calendar object.
COUNTER	Used by an Attendee to negotiate a change in a calendar object. Examples include the request to change the location. Note that COUNTER is little used.
DECLINECOUNTER	Used by the Organizer to decline the proposed counter proposal.

Group scheduling in iTIP is accomplished using the set of “request” and “response” methods described above. The following table shows the methods broken down by who can send them.

Table 3

Originator	Methods
Organizer	PUBLISH, REQUEST, ADD, CANCEL, DECLINECOUNTER
Attendee	REPLY, REFRESH, COUNTER, REQUEST (only when delegating)

Note that for some calendar component types, the allowable methods are a subset of the above set. In addition, apart from timezone components, only one component type is allowed in a single iTIP message.

1. Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IETF RFC 2119, S. BRADNER. *Key words for use in RFCs to Indicate Requirement Levels*. 1997. RFC Publisher. https://www.rfc-editor.org/info/rfc2119.

IETF RFC 4791, C. DABOO, B. DESRUISSEAUX and L. DUSSEAULT. *Calendaring Extensions to WebDAV (CalDAV)*. 2007. RFC Publisher. https://www.rfc-editor.org/info/rfc4791.

IETF RFC 4918, L. DUSSEAULT (ed.). *HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV)*. 2007. RFC Publisher. https://www.rfc-editor.org/info/rfc4918.

IETF RFC 5545, B. DESRUISSEAUX (ed.). *Internet Calendaring and Scheduling Core Object Specification (iCalendar)*. 2009. RFC Publisher. https://www.rfc-editor.org/info/rfc5545.

IETF RFC 5546, C. DABOO (ed.). *iCalendar Transport-Independent Interoperability Protocol (iTIP)*. 2009. RFC Publisher. https://www.rfc-editor.org/info/rfc5546.

IETF RFC 6047, A. MELNIKOV (ed.). *iCalendar Message-Based Interoperability Protocol (iMIP)*. 2010. RFC Publisher. https://www.rfc-editor.org/info/rfc6047.

IETF RFC 8174, B. LEIBA. *Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words*. 2017. RFC Publisher. https://www.rfc-editor.org/info/rfc8174.

IETF RFC 8984, N. JENKINS and R. STEPANEK. *JSCalendar: A JSON Representation of Calendar Data*. 2021. RFC Publisher. https://www.rfc-editor.org/info/rfc8984.

IETF RFC 9073, M. DOUGLASS. *Event Publishing Extensions to iCalendar*. 2021. RFC Publisher. https://www.rfc-editor.org/info/rfc9073.

Internet-Draft draft-ietf-calext-ical-relations-00, MICHAEL DOUGLASS. *Support for Icalendar Relationships*. 2016. https://datatracker.ietf.org/doc/html/draft-ietf-calext-ical-relations-00.

Calendar and scheduling — iCalendar Transport-Independent Interoperability Protocol (iTIP)

2. The Abstract Data Model

This section will describe the classes and attributes required to carry out iTip scheduling. They will be described in abstract terms defined here and later sections will describe how the model is expressed in various concrete representations.

This document will not attempt to provide a full abstract model for the calendar components, properties and parameters defined in IETF RFC 5545. Rather, only those classes and attributes that are required for iTip are described here.

2.1. Calendar Address

Participants in iTip scheduling are identified only by their calendar-address. The calendar-address value type is a uri, usually a mailto.

Example	`mailto:mike@example.org`

2.2. The Owner

when an object is scheduled with iTip one party is the “Owner” (sometimes referred to as the “Organizer”).

The owner (like other participants) is identified by a calendar-address. Additionally, it may have any of: a name; a reference to some form of directory entry; a language specifier.

(Do we drop sentBy?)

If you like PlantUML you may support us!
http://plantuml.com/patreon

```
PlantUML 1.2025.0
[From plantuml20250217-4605-talfoh.pml (line 8) ]

@startuml



'******* IMPORTS ******************************************************
!include /__w/standards.calconnect.org/standards.calconnect.org/src-documents/cc-iTip-rewrite/sources/models/views/../st ...
 cannot include /__w/standards.calconnect.org/standards.calconnect.org/src-documents/cc-iTip-rewrite/sources/models/views/../style.uml.inc
```

If you like PlantUML you may support us!
http://plantuml.com/patreon

Figure 1 — Owner datamodel

2.3. Application Protocol Elements

iTip uses a number of attributes in the calendar objects to provide scheding information and to maintain th scheduling state. These attributes takes different forms in IETF RFC 5545 iCalendar objects and IETF RFC 8984 jsCalendar objects.

The table below lists all iTip attributes and which roperty or parameter is used in each representation.

Table 4

	iCalendar	**jsCalendar**
Organizer	ORGANIZER property	Participant with role owner
Attendee	Attendee property	Participant object
Date stamp	DTSTAMP property	
Method	METHOD property	
Participation status	PARTSTAT parameter	
Status	STATUS property	
Sequence	SEQUENCE property	

iTIP messages are “text/calendar” MIME entities that contain calendaring and scheduling information. The particular type of iCalendar message is referred to as the “method type”. Each method type is identified by a method property specified as part of the “text/calendar” content type. The table below shows various combinations of calendar components and the method types that this specification supports.

Table 5

	Event	Task	Journal	Free-busy
PUBLISH	Yes	Yes	Yes	Yes
REQUEST	Yes	Yes	No	Yes
REFRESH	Yes	Yes	No	No
CANCEL	Yes	Yes	Yes	No
ADD	Yes	Yes	Yes	No
REPLY	Yes	Yes	No	Yes
COUNTER	Yes	Yes	No	No
DECLINECOUNTER	Yes	Yes	No	No

2.3.1. Required properties per components

When sending iTip messages the sender SHOULD send the minimum set of properties. Receivers MUST ignore all optional properties. The tables below show the required properties per component type. All others are optional.

For PUBLISH and REQUEST as many properties as are required to define the component SHOULD be sent and MUST follow the contraints defined in IETF RFC 5545 or IETF RFC 8984.

In particular the sequence property MUST be present if the value is greater than 0.

For REPLY the minimum set of properties should be returned.

1) Required Properties in a VCALENDAR Component

Table 6

Component/Property	Comment
PRODID	
VERSION	Value MUST be 2.0.

1) Required Properties in a VTIMEZONE Component

Table 7

Component/Property	Comment
TZID	

1) Required Properties in a VTIMEZONE STANDARD oor DAYLIGHT Component

Table 8

DTSTART	MUST be local time format.
RDATE	Only if required to specify the transitions
RRULE	Only if required to specify the transitions
TZOFFSETFROM	
TZOFFSETTO	

1) Required Properties in a VALARM Component

Table 9

Component/Property	Comment
ACTION	1

2.3.2. The PUBLISH method

This method is valid for all component types and is an unsolicited posting of an iCalendar object. Any CU may add published components to their calendar. The “ORGANIZER” property MUST be present in a published iCalendar component. “Attendees” MUST NOT be present. Its expected usage is for encapsulating an arbitrary event as an iCalendar object. The “Organizer” may subsequently update (with another “PUBLISH” method), add instances to (with an “ADD” method), or cancel (with a “CANCEL” method) a previously published calendar component.

As many properties as are required to define the component SHOULD be sent and MUST follow the contraints defined in IETF RFC 5546.

In particular the SEQUENCE property MUST be present if the value is greater than 0.

2.3.3. Publishing VFREEBUSY

The “PUBLISH” method for a “VFREEBUSY” calendar component is used to publish busy time data. The method may be sent from one CU to any other. The purpose of the method is to provide a way to send unsolicited busy time data. That is, the busy time data is not being sent as a “REPLY” to the receipt of a “REQUEST” method.

The “ORGANIZER” property MUST be specified in the busy time information. The value is the CU address of the originator of the busy time information.

The busy time information within the iCalendar object MAY be grouped into more than one “VFREEBUSY” calendar component. This capability allows busy time periods to be grouped according to some common periodicity, such as a calendar week, month, or year. In this case, each “VFREEBUSY” calendar component MUST include the “ORGANIZER”, “DTSTART”, and “DTEND” properties in order to specify the source of the busy time information and the date and time interval over which the busy time information covers.

2.3.4. The REQUEST method

This method is valid for

— VEVENT
— VTODO
— VFREEBUSY

The “Organizer” originates the “REQUEST”. The recipients of the “REQUEST” method are the CUs invited to the event, the “Attendees”. For VEVENT and VTODO “Attendees” use the “REPLY” method to convey attendance status to the “Organizer”.

The “UID” and “SEQUENCE” properties are used to distinguish the various uses of the “REQUEST” method. If the “UID” property value in the “REQUEST” is not found on the recipient’s calendar, then the “REQUEST” is for a new calendar component. If the “UID” property value is found on the recipient’s calendar, then the “REQUEST” is for a rescheduling, an update, or a reconfirmation of the “VEVENT” calendar component.

For the “REQUEST” method, multiple “VEVENT” components in a single iCalendar object are only permitted for components with the same “UID” property. That is, a series of recurring events may have instance-specific information. In this case, multiple “VEVENT” components are needed to express the entire series.

2.3.4.1. REQUEST for VEVENT

2.3.5. The REFRESH method

This method is valid for

— VEVENT
— VTODO

2.3.6. The CANCEL method

This method is valid for

— VEVENT
— VTODO
— VJOURNAL

2.3.7. The ADD method

This method is valid for

— VEVENT
— VTODO
— VJOURNAL

2.3.8. The REPLY method

This method is valid for

— VEVENT
— VTODO
— VFREEBUSY

2.3.9. The COUNTER method

This method is valid for

— VEVENT
— VTODO

2.3.10. The DECLINECOUNTER method

This method is valid for

— VEVENT
— VTODO

3. Security Considerations

This specification introduces no new security considerations beyond those identified in IETF RFC 5545.

4. IANA Considerations

4.1. Initialization of the Status registry

This specification updates IETF RFC 5545 by adding a XXXXX value registry to the iCalendar Elements registry and initializing it as per IETF RFC 5545.

5. Acknowledgements

The authors would like to thank the members of the Calendaring and Scheduling Consortium technical committees and the following individuals for contributing their ideas, support and comments:

John Chaffee, Marten Gajda, Ken Murchison

The authors would also like to thank CalConnect, the Calendaring and Scheduling Consortium, for advice with this specification.

Bibliography

[1] OMG BPMN 2.0.2, *Business Process Model and Notation*. 2014. https://www.omg.org/spec/BPMN/2.0.2/About-BPMN.

[2] CalConnect Task Architecture V1.0, Apthorp, A., Daboo, C., Douglass, M., CalConnect, Task Architecture V1.0, http://www.calconnect.org/architectures/Task%20Architecture%201.0.pdf

[3] UN/EDIFACT, D14.A, *UN Economic Commission for Europe, UN/EDIFACT, D14.A, STS STATUS, April 30, 2014,http://www.unece.org/fileadmin/DAM/trade/untdid/d14a/trsd/trsdsts.htm*.

[4] WfRP, Russell, N., ter Hofstede, A.H.M., Edmond, T., van der Aalst,W.M.P., Workflow Resource Patterns, Eindhoven University of Technology, 2004, http://www.workflowpatterns.com/patterns/resource/

[5] OASIS WS-Calendar V1.0, Considine, T., Douglass, M., WS-Calendar Version 1.0, OASIS, 30 July 2011, http://docs.oasis-open.org/ws-calendar/ws-calendar-spec/v1.0/cs01/ws-calendar-spec-v1.0-cs01.html

[6] OASIS WS-HT V1.1, Ings, D., Clement, L., Koenig, D., Mehta, V., Mueller, R., Rangaswamy, R., Rowley, M., Trickovic, I., Web Services — Human Task Version 1.1 (WS-HumanTask), OASIS, 17 August 2010, http://docs.oasis-open.org/ns/bpel4people/ws-humantask/200803