Network Working Group Steve Silverberg, Microsoft
INTERNET-DRAFT Steve Mansour, Netscape
Calendaring and Scheduling Working Group Frank Dawson, Lotus
<ietf-draft-calsch-itip-02.txt> Ross Hopson, ON Technologies
Expires in six months from November 21,1997

iCalendar Transport-Independent Interoperability Protocol (iTIP)
Scheduling Events, BusyTime, To-dos and Journal Entries

Status of this Memo

This document is an Internet-Draft. Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups MAY also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months. Internet-Drafts MAY be updated, replaced, or made obsolete by other documents at any time. It is not appropriate to use Internet-Drafts as reference material or to cite them other than as a "working draft" or "work in progress".

To learn the current status of any Internet-Draft, please check the 1id-abstracts.txt listing contained in the Internet-Drafts Shadow Directories on ds.internic.net (US East Coast), nic.nordu.net (Europe), ftp.isi.edu (US West Coast), or munnari.oz.au (Pacific Rim).

Distribution of this document is unlimited.

Copyright (C) The Internet Society 1997. All Rights Reserved.

Abstract

This document specifies how calendaring systems use iCalendar objects to interoperate with other calendar systems. It does so in a general way so as to allow multiple methods of communication between systems. Subsequent documents specify interoperable methods of communications between systems that use this protocol.

The document outlines a model for calendar exchange that defines both static and dynamic event, to-do, journal and free/busy objects. Static objects are used to transmit information from one entity to another without the expectation of continuity or referential integrity with the original item. Dynamic objects are a superset of static objects and will gracefully degrade to their static counterparts for clients that only support static objects.

Table of Contents

Introduction

This document specifies how calendaring systems use iCalendar objects to interoperate with other calendar systems. In particular, it specifies how to schedule events, to-dos, or daily journal entries. It further specifies how to search for available busy time information. It does so in a general way so as to allow multiple methods of communication between systems. Subsequent documents specify interoperable methods of communications between systems that use this protocol.

This protocol is based on requests sent from an originator and conveyed to one or more recipients. A recipient of a request MAY reply, in order to update their status and MAY also return transaction/request status information. The protocol also supports the ability for the entry originator to modify or cancel the original entry. The elements of the protocol also include the notion of user roles.

1.1 Formatting Conventions

In order to refer to elements of the calendaring and scheduling model, core object or interoperability protocol defined in [ICMS], [ICAL] and [ITIP] several formatting conventions have been utilized.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY" and "OPTIONAL" in this document are to be interopreted as described in [RFC 2119].

Calendaring and scheduling roles defined by [ICMS] are referred to in quoted-strings of text with the first character of each word in upper case. For example, "Organizer" refers to a role of a "Calendar User" within the scheduling protocol defined by [ITIP] Calendar components defined by [ICAL] are referred to with capitalized, quoted-strings of text. All calendar components start with the letter "V". For example, "VEVENT" refers to the event calendar component, "VTODO" refers to the to-do calendar component and "VJOURNAL" refers to the daily journal calendar component. Scheduling methods defined by [ITIP] are referred to with capitalized, quoted-strings of text. For example, "REQUEST" refers to the method for requesting a scheduling calendar

component be created or modified, "REPLY" refers to the method a recipient of a request uses to update their status with the "Organizer" of the calendar component.

Properties defined by [ICAL] are referred to with capitalized, quoted-strings of text, followed by the word "property". For example, "ATTENDEE" property refers to the iCalendar property used to convey the calendar address of a "Calendar User". Property parameters defined by this memo are referred to with lower case, quoted-strings of text, followed by the word "parameter". For example, "value" parameter refers to the iCalendar property parameter used to override the default data type for a property value. Enumerated values defined

by this memo are referred to with capitalized text, either alone or followed by the word "value".

In tables, the quoted-string text is specified without quotes in order to minimize the table length.

1.2 Related Documents

Implementers will need to be familiar with several other memos that, along with this one, describe the Internet calendaring and scheduling standards. This document, [ITIP], specifies an interoperability protocol for scheduling between different implementations;

[ICMS] - describes the abstract model and defines common terms and concepts;

[ICAL] - specifies the objects, data types, properties and property parameters used in the protocols, along with the methods for representing and encoding them;

[IRIP] - specifies an Internet real time protocol binding for [ITIP].

[IMIP] specifies an Internet email binding for [ITIP].

This memo does not attempt to repeat the specification of concepts or definitions from these other memos. Where possible, references are made to the memo that provides for the specification of these concepts or definitions.

1.3 Calendar Roles

Roles are a behavior or set of activities performed by particular groups of users or agents at a given state of the calendar transaction. This specification describes 4 roles that determine a range of actions and responsibilities specific to each role.

Role Name	Description
Owner	The calendar entry owner is the only Calendar User allowed to directly modify an entry using the iTIP protocol. However, the Owner MAY delegate or assign an Organizer to manage the entry on their behalf. Usually, a calendar entry Owner is also the Organizer.
Organizer	The Organizer controls manipulation of the calendar entry. In most cases, the Owner and the Organizer are the same Calendar User.
Attendee	An Attendee is a Calendar User associated with a calendar entry via a Request method issued by an Organizer or another Attendee. Attendees are not capable of directly manipulating calendar entries, but MUST act through the Organizer.
Delegate	A Delegate is a proxy that acts on behalf of another Calendar User. ITIP addresses two forms of delegation: 1) An Owner MAY delegate or re-assign an Organizer to manage a calendar entry 2) An Attendee MAY delegate a calendar entry request to another Calendar User.

1.4 iTIP Transactions

This protocol defines seven methods for exchanging [ICAL] objects for

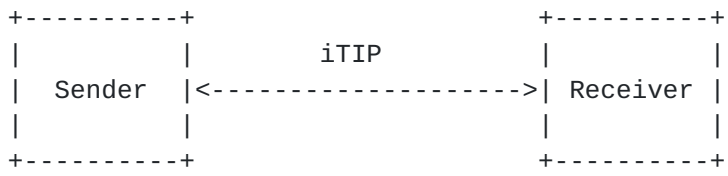
the purposes of group calendaring and scheduling between "Calendar Users". The methods are defined below and their usage and semantics are outlined in section 3 of this document.

Method	Description
PUBLISH	Used to publish a calendar entry 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 entry with other Calendar Users. Requests are interactive in that they MAY require the receiver to respond using the the Reply methods. Meeting Requests, Busy Time requests and the assignment of VTODOs to other Calendar Users are all examples. Requests are also used by the "Organizer" to update the status of a calendar entry.
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.
CANCEL	The Cancel method is used to cancel an existing calendar entry such as a VEVENT or VTODO.
REFRESH	The Refresh method is used by an "Attendee" to request the latest version of a calendar entry
COUNTER	The Counter method is used by an "Attendee" to negotiate a change in the calendar entry. Examples include the request to change a proposed Event time or change the due date for a VTODO.
DECLINE-COUNTER	Used by the "Organizer" to decline the proposed counter-proprosal by an "Attendee"

2 Interoperability Models

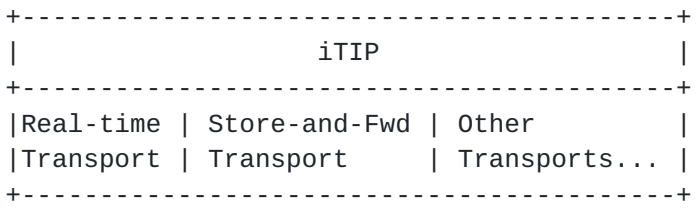
There are two distinct protocols relevant to interoperability: an "Application Protocol" and a "Transport Protocol". The Application Protocol defines the content of the iCalendar objects sent between sender and receiver to accomplish the scheduling transactions listed in section 1.4. The Transport Protocol defines how the iCalendar objects are sent between the sender and receiver. This document focuses on the Application Protocol.

The connection between Sender and Receiver in the diagram below refers to the Application Protocol. In particular, the iCalendar objects passed from the Sender to the Receiver which conform to those presented in Section 3, Application Protocol Elements.



There are several variations of this diagram in which the Sender and Receiver take on various roles of "CUA" or CS. These variants are detailed in the Model document [ICMS]

The architecture of iTIP is depicted in the diagram below. An application written to this specification MAY work with bindings for the store-and-forward transport, the real time transport, or both. Also note that iTIP could be bound to other transports. If a capability is not available on a particular transport binding, iTIP provides a mechanism for indicating so.



2.1 Application Protocol

The model for the application protocol is centered with the "Organizer" of the calendar entry. That is, the "Organizer" of a

Calendar entry sends a request to one or more "Attendees". The "Attendees" then reply to the "Organizer". The "Organizer" maintains the status of the event.

The "Owner" is usually the "Organizer" of the calendar entry. However, the "Owner" MAY delegate or assign an "Organizer" to manage the calendar entry on their behalf. In cases where the "Owner" has delegated to another "Organizer", the "Owner" must still be specified in associated "REQUEST" and "COUNTER" methods.

The data sources for the application protocol are the "Calendar Users". Examples of these users are the "Organizer" and "Attendees" of an iCalendar event. The data objects are the iCalendar objects that are exchanged between "Calendar Users".

2.1.1 Calendar Entry State

There are two distinct states relevant to calendar entries: the overall state of the entry and the state associated with an "Attendee" to that entry.

The state of an entry is defined by the "STATUS" property and is controlled by the "Organizer." There is no default value for the "STATUS" property. The "Organizer" MAY either set the "STATUS" property to TENTATIVE or CONFIRMED values. The "Organizer" MAY also set the "STATUS" property to CANCELLED value by sending a "CANCEL" method to each "Attendee".

The state of a particular "Attendee" relative to an entry is defined by the "STATUS" property parameter in the "ATTENDEE" property for that "Attendee". When an "Organizer" sends out an entry, the state associated with each "Attendee" is NEEDS-ACTION. Each "Attendee" MAY modify their "ATTENDEE" property "STATUS" parameter to an appropriate value and send it back to the "Organizer" in a "REPLY" message.

3 Application Protocol Elements

Messages are "text/calendar" MIME entities that contain calendaring and scheduling information. The particular type of [ICAL] 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 memo supports.

	VEVENT	VTODO	VJOURNAL	VFREEBUSY
Publish	Yes	Yes	Yes	Yes

```
          |Request  |  Yes   |  Yes  |  No      |   Yes     |
          |Refresh  |  Yes   |  Yes  |  Yes     |   No      |
          |Cancel   |  Yes   |  Yes  |  Yes     |   No      |
          |Reply    |  Yes   |  Yes  |  No      |   Yes     |
          |Counter  |  Yes   |  Yes  |  No      |   No      |
```



```
          |Decline- |        |       |          |           |
          |Counter  |  Yes   |  Yes  |  No      |   No      |
          +=============================================+
```

Each method type is defined in terms of its associated properties. Some properties are required, some are optional and others are excluded. The property restrictions are expressed in this memo using the following formal notation:

```
  prop-restriction    = "(" description component method
            1*MUST-component *MAY-component *not-component ")"

  description    = "DESCRIPTION" *ws text
  component      = "COMPONENT" *ws ("CALPROPOS" / "VEVENT" /
                   "VTODO" / "VJOURNAL")
  method         = "METHOD" *ws <any of the methods defined in this
                    memo for the associated calendar component>

  MUST-component = "MUST COMPONENT =" *ws ("VEVENT" / "VTODO" /
                   "VJOURNAL" / "VTIMEZONE" / "VALARM" /
                   "VFREEBUSY" / "X-TOKEN")
                   *ws "(" [ws] 1*MUST-props *ws
                   *MAY-props *ws *not-props *ws ")"

  MAY-component  = "MAY COMPONENT =" *ws ("VEVENT" / "VTODO" /
                   "VJOURNAL" / "VTIMEZONE" / "VALARM" /
                   "VFREEBUSY" / "X-TOKEN")
                   *ws "(" [ws] ((*MUST-props *ws
                   *MAY-props *ws *not-props *ws) / any) ")"

  not-component  = "NOT COMPONENT =" *ws ("VEVENT" / "VTODO" /
                   "VJOURNAL" / "VTIMEZONE" / "VALARM" /
                   "VFREEBUSY" /"X-TOKEN")

  MUST-props     = "MUST PROPERTY =" *ws restriction-list
  MAY-props      = "MAY PROPERTY =" *ws (restriction-list / any)
  not-props      = "NOT PROPERTY =" *ws (restriction-list)
```

```
restriction-list    = restriction
                    / restriction *ws "," *ws restriction
restriction    = property-name
                 [*ws "{" parm-or-val-restriction "}"]
property-name  = <any of the valid properties for the component>
parm-or-val-restriction  = <a text string description of a
               constraint on the property parameters
               or values>
any       = "ANY"  -- Specifies that any permissible
            properties are allowed - -

ws        = HTAB / SPACE

HTAB      = <Horizontal TAB character>
SPACE     = <Required Space character>
```


3.1 Methods For "VEVENT" Calendar Component

This section defines the property set restrictions for the method types that are applicable to the "VEVENT" calendar component. Each of the methods is defined using a grammar that clarifies the property constraints that define the particular method.

The following summarizes the methods that are defined for the "VEVENT" calendar component.

Method	Description
PUBLISH	Post notification of an event. Used primarily as a method of advertising the existence of an event.
REQUEST	Make a request for an event. This is an explicit invitation to one or more "Attendees". Event Requests are also used to update or change an existing event. Clients that cannot handle REQUEST MAYdegrade the event to view it as an PUBLISH.
REPLY	Reply to an event request. Clients MAYset their status to

	ACCEPTED, DECLINED, TENTATIVE, DELEGATED.
CANCEL	Cancel an existing event request.
REFRESH	A request sent to an by an "Attendee" "Organizer" asking for the latest version of an event to be resent to the requester.
COUNTER	Counter a REQUEST with an alternative proposal.
DECLINECOUNTER	Decline a counter proposal by an "Attendee".

3.1.1 PUBLISH

The "PUBLISH" method in a "VEVENT" calendar component provides an unsolicited posting of an iCalendar object. Any "Calendar User" MAY add the published components to their calendar. It requires and accepts no responses to the "Organizer". Its expected usage is for encapsulating an arbitrary event or to-do as an iCalendar object. The "Organizer" MAY subsequently update (with another "PUBLISH" method) or cancel (with a "CANCEL" method) a previously published "VEVENT" calendar component.

This method type is an iCalendar object that conforms to the following property constraints:

(DESCRIPTION "Event - Publish" COMPONENT "VEVENT" METHOD "PUBLISH


```
MUST COMPONENT = CALPROPS (
     MUST PROPERTY = PRODID, VERSION{"2.0"}, METHOD{"PUBLISH"})
MUST COMPONENT = VEVENT (
     MUST PROPERTY = ATTENDEE{ROLE=OWNER and ORGANIZER if
          different}, DESCRIPTION{MAY BE NULL},DTSTAMP, DTSTART,
          SEQUENCE{IF NE 0}, UID
     MAY PROPERTY = ATTACH, ATTENDEE{other than ROLE=OWNER |
          ORGANIZER}, CATEGORIES, CLASS, COMMENT,
          CREATED, DTEND, EXDATE, EXRULE, GEO, LAST-MODIFIED,
          LOCATION,PRIORITY, RELATED-TO, RDATE, RECURRENCE-ID,
          RESOURCES, RRULE, SEQUENCE{IF EQ 0},
          STATUS{TENTATIVE/CONFIRMED/CANCELLED},
          SUMMARY{MAYBE NULL}, URL
     NOT PROPERTY = REQUEST-STATUS, TRANSP)
MAY COMPONENT = VTIMEZONE (
```

```
          MUST PROPERTY = DTSTART, TZOFFSET
          MAY PROPERTY = COMMENT, DAYLIGHT, (RDATE / RRULE), TZNAME,
          NOT PROPERTY = CREATED)
     MAY COMPONENT = VALARM (
          MUST PROPERTY = CATEGORIES, DSTART, DURATION, REPEAT
          MAY PROPERTY = ATTACH, DESCRIPTION, SUMMARY
          NOT PROPERTY = COMMENT, CREATED, LAST-MODIFIED, RELATED-TO,
               URL)
     MAY COMPONENT = X-TOKENS (ANY)
     NOT COMPONENT = VTODO
     NOT COMPONENT = VJOURNAL
     NOT COMPONENT = VFREEBUSY
     )
```

3.1.2 REQUEST

The "REQUEST" method in a "VEVENT" calendar component provides the following scheduling functions:

- Invite "Attendees" to an event;
- Reschedule an existing event;
- Update the details of an existing event, without rescheduling it;
- Update the status of "Attendees" of an existing event, without rescheduling it;
- Reconfirm an existing event, without rescheduling it;
- For an existing "VEVENT" calendar component, delegate the role of "Attendee" to another "Calendar User";
- For an existing "VEVENT" calendar component, delegate the role of "Organizer" to another "Calendar User".

The originator of the "REQUEST" method is the "Organizer" of the event. Normally this is the "Owner" of the event. The recipient of the "REQUEST" method is the "Calendar User" invited to the event, called the "Attendee". The "Attendee" uses the "REPLY" method to convey their attendance status to the "Organizer" of a VEVENT

"REQUEST".

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 "VEVENT" calendar component. If the "UID" property value is found on the recipient's calendar, then the "REQUEST"' is for either a rescheduling, an update, or a reconfirm of the "VEVENT" calendar component.

If the "Organizer" of the "REQUEST" method is not authorized to make an event request on the "Attendee's" calendar system, then an exception is returned in the "REQUEST-STATUS" property of a subsequent "REPLY" method, but no scheduling action is performed.

This method type is an iCalendar object that conforms to the following property constraints:

```
  (DESCRIPTION "Event - Request" COMPONENT "VEVENT" METHOD "REQUEST"

  MUST COMPONENT = CALPROPS (
       MUST PROPERTY = PRODID, VERSION{"2.0"}, METHOD{"REQUEST"})
  MUST COMPONENT = VEVENT (
       MUST PROPERTY = ATTENDEE{ ROLE=OWNER and ORGANIZER if
            different and "STATUS parameter absent or
            STATUS=NEEDS-ACTION on Attendees}, DESCRIPTION{MAYBE
            NULL}, DTSTAMP,DTSTART,SEQUENCE{IF NE 0}, UID
       MAY PROPERTY = ATTACH, CATEGORIES, CLASS, COMMENT, CREATED,
            DTEND, EXDATE, EXRULE, GEO, LAST-MODIFIED, LOCATION,
            PRIORITY, RDATE, RECURRENCE-ID, RELATED-TO, RESOURCES,
            RRULE, SEQUENCE{IF EQ 0}, STATUS{TENTATIVE/CONFIRMED
            /CANCELLED}, SUMMARY {MAYBE NULL}, TRANSP, URL
       NOT PROPERTY = REQUEST-STATUS)
  MAY COMPONENT = VTIMEZONE (
       MUST PROPERTY = DTSTART, TZOFFSET
       MAY PROPERTY = COMMENT, DAYLIGHT, (RDATE / RRULE), TZNAME
       NOT PROPERTY = CREATED)
  MAY COMPONENT = VALARM (
       MUST PROPERTY = CATEGORIES, DSTART, DURATION, REPEAT
       MAY PROPERTY = ATTACH, DESCRIPTION, SUMMARY
       NOT PROPERTY = COMMENT, CREATED, LAST-MODIFIED, RELATED-TO,
            URL)
  MAY COMPONENT = X-TOKENS (ANY)
  NOT COMPONENT = VTODO
  NOT COMPONENT = VJOURNAL
  NOT COMPONENT = VFREEBUSY
  )
```

3.1.2.1 REQUEST for Rescheduling an Event

The "REQUEST" method MAY be used to reschedule an event.

A rescheduled event involves a change to the existing event in terms of it's time or recurrence intervals and possibly the location or description. If the recipient "CUA" of a "REQUEST" method finds that the "UID" property value already exists on the calendar, but that the "SEQUENCE" property value in the "REQUEST" method is greater than the value for the existing event, then the "REQUEST" method describes a rescheduling of the event.

3.1.2.2 REQUEST for Update or Reconfirmation of an Event

The "REQUEST" method MAY be used to update or reconfirm an event.

An update to an existing event does not involve changes to the time or recurrence intervals, and might not involve a change to the location or description for the event. If the recipient "CUA" of a "REQUEST" method finds that the "UID" property value already exists on the calendar and that the "SEQUENCE" property value in the "REQUEST" is the same as the value for the existing event, then the "REQUEST" method describes an update of the event details, but no rescheduling of the event.

The update "REQUEST" method is the appropriate response to a "REFRESH" method sent from an "Attendee" to the "Organizer" of an event.

Unsolicited "REQUEST" methods MAY also be sent by the "Organizer" of an event. The unsolicited "REQUEST" methods MAY be used to either update the details of the event, without rescheduling it, to update the "STATUS" property parameter of "Attendees", or to reconfirm the event.

3.1.2.3 REQUEST for Delegating an Event from an "Attendee" to another CU

The "REQUEST" method MAY be used to delegate an event to another "Calendar User". The method is used to delegate the "Attendee's" role (i.e., "Organizer" or "Attendee") for an event. The "REQUEST" method for delegation is sent by one of the "Attendees" of an existing event

request to some other "Calendar User". In order to avoid scheduling loops, the method MUST NOT be sent from an "Attendee" back to the "Organizer" of the event. An "Attendee" MAY NOT delegate to the "Organizer" of the event.

For the purposes of this description, the "Attendee" delegating the event is referred to as the "delegator". The "Attendee" receiving the delegation request is referred to as the "delegatee".

The "delegator" of an event MUST forward the existing "REQUEST" method for an event to the "delegatee". The event description MUST include the "delegator's" up-to-date event definition. The "REQUEST" method MUST also include an "ATTENDEE" property with the calendar address of the "delegatee". The "delegator" MUST also send a "REPLY" method back to the "Organizer" with the "delegator's" "Attendee" property "STATUS" parameter value set to DELEGATED. In addition, the "DELEGATED-TO" parameter SHOULD be included with the calendar address of the "delegatee". A response to the delegation "REQUEST" is sent from the "delegatee" to the "Organizer" and optionally, to the "delegator". The "REPLY" method from the "delegatee" SHOULD include their "ATTENDEE" property with the "DELEGATED-FROM" parameter value of the "delegator's"calendar address.

The delegation "REQUEST" method MUST assign the values of the "RSVP" and "EXPECT" property parameters associated with the "delegator's" "ATTENDEE" property to that of the "delegatee's" "ATTENDEE" property. For example if the "delegator's" "ATTENDEE" property specifies "RSVP=TRUE;EXPECT=REQUEST", then the "delegatee's" "ATTENDEE" property MUST specify "RSVP=TRUE;EXPECT=REQUEST".

3.1.2.4 REQUEST for Delegating role of "Organizer" to another CU

If the "Owner" of an existing "VEVENT" calendar component wishes to delegate the role of "Organizer" to another CU, they MAY issue another "REQUEST" method that notifies all "Attendees" and the new "Organizer" of this change. The "Owner MUST modify several property parameters of the "ATTENDEE" property including the "ROLE", where the role of "Owner" and "Organizer" MUST be specified. Additionally, the "DELEGATED-TO" and "DELEGATED-FROM" parameters MUST specify the "Organizer" and "Owner" calendar addresses. The "Owner" MAY request reconfirmation by incrementing the "SEQUENCE" property and setting the "RSVP" property parameter to TRUE. This will cause a

reconfirmation to be sent to the new organizer.

3.1.2.5 REQUEST for Changing the "Organizer" from one CU to another

An "Owner" of an existing "VEVENT" calendar component MAY change the "Organizer" from one "CU" to another by sending a "REQUEST" method. The "ROLE" property parameter value of ORGANIZER MUST be assigned to the new "Organizer". If the old "Organizer" is still an "Attendee" then "ROLE" property parameter for that "CU" MUST be set to ATTENDEE.

3.1.2.6 REQUEST for Changing the "Owner"

This memo does not support the notion of changing ownership of a "VEVENT" calendar component.

3.1.2.7 REQUEST Forwarded To An Uninvited Calendar User

An "Attendee" invited to an event MAY invite another uninvited "Calendar User" to the event. The invited "Attendee" accomplishes this scheduling action by forwarding the original "REQUEST" method to the uninvited "Calendar User". The forwarded "REQUEST" method need not include a new "ATTENDEE" property for the uninvited "Attendee". Whether the uninvited "Calendar User" is added to the attendee list, and thus informed of changes to the "VEVENT" calendar component is an implementation issue.

3.1.2.8 REQUEST Updated Attendee Status

An "Organizer" of an event MAY also request an updated status from one of the "Attendees". This is achieved by the "Organizer" sending a "REQUEST" method to the "Attendee" with the "ATTENDEE;RSVP=TRUE" property sequence. The "SEQUENCE" property for the event is not changed from its previous value. A recipient will determine that the only change in the "REQUEST" is that their "RSVP" property parameter indicates a request for an updated status. The recipient SHOULD respond with a "REPLY" method indicating their current status with respect to the "REQUEST".

This capability MAY also be achieved by the "Organizer" sending the "REFRESH" method to the "Attendees".

3.1.3 REPLY

The "REPLY" method in a "VEVENT" calendar component is used to respond (e.g., accept or decline) to a request or to reply to a delegation request. When used in to provide a delegation response, the "delegator" SHOULD include the calendar address of the "delegatee" on the "DELEGATED-TO" property parameter of the "delegator's" "ATTENDEE" property. The "delegatee" SHOULD include the calendar address of the "delegator" on the "DELEGATED-FROM" property parameter of the "delegatee's" "ATTENDEE" property.

The "REPLY" method MAY also be used to respond to an unsuccessful "REQUEST" method. Depending on the value of the "REQUEST-STATUS" property no scheduling action MAY have been performed.

The "Organizer" of an event MAY receive the "REPLY" method from a "Calendar User" not in the original "REQUEST". For example, a "REPLY" method MAY be received from a "delegatee" to an event. In addition, the "REPLY" method MAY be received from an unknown "Calendar User", forwarded the "REQUEST" from an invited "Attendee". This uninvited "Attendee" MAY be accepted, or the "Organizer" MAY cancel the event for the uninvited "Attendee" by sending them a "CANCEL" method to the univited "Attendee".

This method type is an iCalendar object that conforms to the following property constraints:

```
  (DESCRIPTION "Event - Reply" COMPONENT "VEVENT" METHOD "REPLY"

  MUST COMPONENT = CALPROPS (
       MUST PROPERTY = PRODID, VERSION{"2.0"}, METHOD{"REPLY"})
  MUST COMPONENT = VEVENT (
       MUST PROPERTY = ATTENDEE{MUST be address of ATTENDEE
            replying}, DTSTAMP, SEQUENCE{IF NE 0}, UID{UID
            associated with original REQUEST}
       MAY PROPERTY = COMMENT{provides comment from ATTENDEE to
            "Organizer"}, EXDATE, EXRULE, RECURRENCE-ID, REQUEST-
```

```
            STATUS, SEQUENCE{IF EQ 0}, SUMMARY {MAYBE NULL}, URL
       NOT PROPERTY = ATTACH, CATEGORIES, CLASS, CREATED,
            DESCRIPTION, DTSTART, DTEND, GEO, LAST-MODIFIED,
            PRIORITY, RELATED-TO, RESOURCES, RDATE, RRULE, STATUS,
            SUMMARY, TRANSP, URL)
  MAY COMPONENT = X-TOKENS (ANY)
  NOT COMPONENT = VTODO
  NOT COMPONENT = VJOURNAL
  NOT COMPONENT = VFREEBUSY
  NOT COMPONENT = VTIMEZONE
  NOT COMPONENT = VALARM
  )
```

3.1.4 CANCEL

The "CANCEL" method in a "VEVENT" calendar component is used to send a cancellation notice of an existing event request to the "Attendees". The message is sent by the "Organizer" of an event to the "Attendees" of the event. For a recurring event, either the whole event or instances of an event MAY be cancelled. To cancel the complete range of recurring event, the "UID" property value for the event MUST be specified in the "CANCEL" method. In order to cancel an individual instance of the event, the "RECURRENCE-ID" property value for the event MUST be specified in the "CANCEL" method. In order to cancel a sequence of recurring events, either the"RECURRENCE-ID" property for the first event instance in the sequence MUST be specified with the "RANGE" property parameter value of THISANDPRIOR or THISANDFUTURE to indicate cancellation of the event instances before and after the first event instance, respectively. Lastly, individual recurrence instances MAY be cancelled by specifying multiple"RECURRENCE-ID" properties corresponding to the instances to be cancelled.

This method type is an iCalendar object that conforms to the following property constraints:

```
  (DESCRIPTION "Event - Cancel" COMPONENT "VEVENT" METHOD "CANCEL"

  MUST COMPONENT = CALPROPS (
       MUST PROPERTY = PRODID, VERSION{"2.0"}, METHOD{"CANCEL"})
```



```
  MUST COMPONENT = VEVENT (
       MUST PROPERTY = DTSTAMP, SEQUENCE{IF NE 0}, UID{UID
```

```
            associated with original REQUEST}
       MAY PROPERTY = COMMENT{provides comment from "Organizer" to
            ATTENDEE}, EXDATE, EXRULE, RECURRENCE-ID, SEQUENCE{IF EQ
            0}, STATUS{CANCELLED}
       NOT PROPERTY = ATTACH, ATTENDEE, CATEGORIES, CLASS, CREATED,
            DESCRIPTION, DTSTART, DTEND, GEO, LAST-MODIFIED,
            PRIORITY, RDATE, RELATED-TO, RESOURCES, REQUEST-STATUS,
            RRULE, SUMMARY, TRANSP, URL)
  MAY COMPONENT = X-TOKENS (ANY)
  NOT COMPONENT = VTODO
  NOT COMPONENT = VJOURNAL
  NOT COMPONENT = VFREEBUSY
  NOT COMPONENT = VTIMEZONE
  NOT COMPONENT = VALARM
  )
```

3.1.5 REFRESH

The "REFRESH" method in a "VEVENT" calendar component is used by "Attendees" of an existing event to request an updated description from the event "Organizer". The "REFRESH" method MUST specify the "UID" property corresponding to the event needing update. A recurrence instance of an event MAY be requested by specifying the "RECURRENCE-ID" property corresponding to the associated event. The "Organizer" MUST respond with the latest description and version of the event. This method is intended to facilitate machine processing of event update requests by an "Attendee".

This method type is an iCalendar object that conforms to the following property constraints:

```
  (DESCRIPTION "Event - Refresh" COMPONENT "VEVENT" METHOD "REFRESH"

  MUST COMPONENT = CALPROPS (
       MUST PROPERTY = PRODID, VERSION{"2.0"}, METHOD{"REFRESH"})
  MUST COMPONENT = VEVENT (
            MUST PROPERTY = ATTENDEE{address of requestor}, DTSTAMP,
            SEQUENCE{IF NE 0}, UID{UID associated with original
            REQUEST}
       MAY PROPERTY = COMMENT{provides comment from ATTENDEE to
            "Organizer"}, RECURRENCE-ID, SEQUENCE{IF EQ 0}
       NOT PROPERTY = ATTACH, CATEGORIES, CLASS, CREATED,
            DESCRIPTION, DTSTART, DTEND, EXDATE, EXRULE, GEO,
            LAST-MODIFIED, PRIORITY, RDATE,RELATED-TO, RESOURCES,
            REQUEST-STATUS, RRULE,SUMMARY, STATUS, TRANSP, URL)
  MAY COMPONENT = X-TOKENS (ANY)
  NOT COMPONENT = VTODO
  NOT COMPONENT = VJOURNAL
  NOT COMPONENT = VFREEBUSY
  NOT COMPONENT = VTIMEZONE
```

NOT COMPONENT = VALARM

)

3.1.6 COUNTER

The "COUNTER" method in a "VEVENT" calendar component is used by an "Attendee" of an existing event to submit to the "Organizer" a counter proposal to the event description. The "Attendee" MUST send the message to the "Organizer" of the event.

The counter proposal is an iCalendar object consisting of a VEVENT calendar component describing the complete description of the alternate event.

The "Organizer" rejects the counter proposal by sending the "Attendee" a VEVENT "DECLINE-COUNTER" method. The "Organizer" accepts the counter proposal by sending all of the "Attendees" to the event a VEVENT "REQUEST" method rescheduling the event. In the later case, the "Organizer" SHOULD reset the individual "RSVP" property parameter values to TRUE on each "ATTENDEE" property in order to force a response by the "Attendees".

This method type is an iCalendar object that conforms to the following property constraints:

(DESCRIPTION "Event - Counter Proposal" COMPONENT "EVENT"
METHOD "COUNTER"

MUST COMPONENT = CALPROPS (
MUST PROPERTY = PRODID, VERSION{"2.0"}, METHOD{"COUNTER"})
MUST COMPONENT = VEVENT (
MUST PROPERTY = ATTENDEE{STATUS parameter absent or
STATUS=NEEDS ACTION, Owner and Organizer if different,
MAY also be used to propose other "Attendees"},
DESCRIPTION{MAYBE NULL}, DTSTAMP, DTSTART,
SEQUENCE{IF NE 0}, UID{MUST be the UID associated with
the REQUEST being countered}
MAY PROPERTY = ATTACH, CATEGORIES, CLASS, COMMENT{provides a
comment from the ATTENDEE to the "Organizer"}, CREATED,
DTEND, EXDATE, EXRULE, GEO, LAST-MODIFIED, LOCATION,
PRIORITY, RDATE, RECURRENCE-ID, RELATED-TO, RESOURCES,
RRULE, SEQUENCE{IF EQ 0}, STATUS{TENTATIVE/CONFIRMED/

```
            CANCELLED} , SUMMARY {MAYBE NULL}, TRANSP, URL
          NOT PROPERTY = COMPLETED, DUE, DURATION, REQUEST-STATUS)
     MAY COMPONENT = VTIMEZONE (
          MUST PROPERTY = DTSTART, TZOFFSET
          MAY PROPERTY = COMMENT, DAYLIGHT, (RDATE / RRULE), TZNAME
          NOT PROPERTY = CREATED)
     MAY COMPONENT = VALARM (IF COMPONENT EXISTS
          MUST PROPERTY = CATEGORIES, DSTART, DURATION, REPEAT
          MAY PROPERTY = ATTACH, DESCRIPTION, SUMMARY
     NOT PROPERTY = COMMENT, CREATED, LAST-MODIFIED, RELATED-TO, URL)
     MAY COMPONENT = X-TOKENS (ANY)
     NOT COMPONENT = VTODO
```



```
     NOT COMPONENT = VJOURNAL
     NOT COMPONENT = VFREEBUSY
     )
```

3.1.7 DECLINECOUNTER

The "DECLINE-COUNTER" method in a "VEVENT" calendar component is used by the "Organizer" of an event to reject a counter proposal submitted by an "Attendee". The "Organizer" MUST send the "DECLINE-COUNTER" message to the "Attendee" that sent the "COUNTER" method to the "Organizer".

The counter proposal is an iCalendar object consisting of a VEVENT calendar component describing the complete description of the alternate event.

The "Organizer" rejects the counter proposal by sending the "Attendee" a "DECLINE-COUNTER" method. The "Organizer" accepts the counter proposal by sending all of the "Attendees" to the event a "REQUEST" method; rescheduling the event. Since this is a rescheduled event, the "SEQUENCE" property value will be incremented. In the later case, the "Organizer" SHOULD reset the individual "RSVP" parameter values to TRUE on all of the "ATTENDEE" properties; in order to force a response by the "Attendees".

This method type is an iCalendar object that conforms to the following property constraints:

```
  (DESCRIPTION "Event - Cancel" COMPONENT "VEVENT" METHOD
       "DECLINECOUNTER"
```

```
MUST COMPONENT = CALPROPS (
     MUST PROPERTY = PRODID, VERSION{"2.0"},METHOD{
     "DECLINECOUNTER"})
MUST COMPONENT = VEVENT (
     MUST PROPERTY = DTSTAMP, SEQUENCE{IF NE 0}, UID{same UID
          specified In Original REQUEST and subsequent COUNTER}
     MAY PROPERTY = COMMENT{provides comment from "Organizer" to
          ATTENDEE}, RECURRENCE-ID, REQUEST-STATUS, SEQUENCE{IF EQ
          0}
     NOT PROPERTY = ATTACH, ATTENDEE, CATEGORIES, CLASS, CREATED,
          DESCRIPTION, DTSTART, DTEND, EXDATE, EXRULE, GEO,
          LAST-MODIFIED, PRIORITY, RDATE, RELATED-TO, RESOURCES,
          RRULE,STATUS, SUMMARY, TRANSP, URL)
MAY COMPONENT = X-TOKENS (ANY)
NOT COMPONENT = VTODO
NOT COMPONENT = VJOURNAL
NOT COMPONENT = VFREEBUSY
NOT COMPONENT = VTIMEZONE
NOT COMPONENT = VALARM
)
```


3.2 Methods For VFREEBUSY Component

This section defines the property set for the methods that are applicable to the "VFREEBUSY" calendar component. Each of the methods is defined using a grammar that specifies the property constraints that define the particular method.

This memo only addresses the transfer of busy time information. Applications desiring free time information MUST infer this from available 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 "ATTENDEE", "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.

The "FREEBUSY" property value MAY include a list of values, separated by the COMMA character (ASCII decimal 44). Alternately, multiple busy time periods MAY be specified with multiple instances of the "FREEBUSY" property. Both forms MUST be supported by implementations conforming to this document. Duplicate busy time periods SHOULD NOT be specified in an iCalendar object. However, two different busy time periods MAY overlap.

"FREEBUSY" properties SHOULD be sorted such that their values are in ascending order, based on the start time, and then the end time, with the earliest periods first. For example, today's busy time information SHOULD appear after yesterday's busy time information. And the busy time for this half hour SHOULD appear after the busy time for earlier today.

Since events MAY span a day boundary, free busy time period MAY also span a day boundary. Individual "A" requests busy time from individuals "B", "C" and "D". Individual "B" and "C" replies with busy time data to individual "A". Individual "D" does not support busy time requests and does not reply with any data. If the transport binding supports exception messages, then a "unsupported capability" message is returned by individual "D" to individual "A".

The following summarizes the methods that are defined for the "VFREEBUSY" calendar component.

Method	Description
PUBLISH	Publish unsolicited busy time data.
REQUEST	Request busy time data.
REPLY	Reply to a busy time request.

3.2.1 PUBLISH

The "PUBLISH" method in a "VFREEBUSY" calendar component is used to publish busy time data. The method MAY be sent from one "Calendar User" to any other. The purpose of the method is to provide a message for sending unsolicited busy time data. That is, the busy time data

is not being sent as a "REPLY" to the receipt of a "REQUEST" method.

Busy time intervals are represented by individual instances of the "FREEBUSY" property. There is one occurrence of the property for each busy time interval. Duplicate busy time periods SHOULD NOT be returned. However, two different busy time periods MAY overlap.

The "FREEBUSY" property value MAY include a list of values, separated by the COMA character (ASCII decimal 44).

"FREEBUSY" properties SHOULD be sorted such that their values are in ascending order, from the most recent to past. For example, today's busy time information SHOULD appear after yesterday's busy time information. And the busy time for this half hour SHOULD appear after the busy time for earlier today.

Since events MAY span a day boundary, free busy time period MAY also span a day boundary.

The busy time periods 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 "ATTENDEE", "DTSTART" and "DTEND" properties in order to specify the originator and date and time interval for the busy time information.

The "ATTENDEE" property MUST be specified in the busy time information. The value is the "Calendar User" address of the originator of the busy time information.

This method type is an iCalendar object that conforms to the following property constraints:

```
  (DESCRIPTION "Busy - Busy Time Publish" COMPONENT "VFREEBUSY"
       METHOD "PUBLISH"

  MUST COMPONENT = CALPROPS (
       MUST PROPERTY = PRODID, VERSION{"2.0"}, METHOD{"PUBLISH"})
  MUST COMPONENT = VFREEBUSY (
       MUST PROPERTY = ATTENDEE{address of originator of busy time
            data},FREEBUSY{values MUST all be of the same data type.
            Multiple instances are allowed. Multiple instances MUST
            be sorted in ascending order. Values MAY NOT overlap},
            RELATED-TO{refers to another related VFREEBUSY
```

```
            component},
       MAY PROPERTY = COMMENT{comment from attendee to originator of
            request}, CREATED{specifies when the busy time data was
            created}, DTSTART{represents start of interval for busy
            time data}, DTEND{represents end of interval for busy
            time data},LAST-MODIFIED{specifies when busy time data
            was last modified}, SEQUENCE{IF EQ 0}, URL{specifies busy
            time URL}
       NOT PROPERTY = DTSTAMP, DURATION, REQUEST-STATUS, SEQUENCE,
            UID)
  MAY COMPONENT = VTIMEZONE (
       MUST PROPERTY = DTSTART, TZOFFSET
       MAY PROPERTY = COMMENT, DAYLIGHT, (RDATE / RRULE), TZNAME
       NOT PROPERTY = CREATED)
  MAY COMPONENT = X-TOKENS (ANY)
  NOT COMPONENT = VEVENT
  NOT COMPONENT = VTODO
  NOT COMPONENT = VJOURNAL
  NOT COMPONENT = VALARM
  )
```

3.2.2 REQUEST

The "REQUEST" method in a "VFREEBUSY" calendar component is used to ask a "Calendar User" for their busy time information. The request MAY be for a busy time information bounded by a specific date and time interval.

This message only permits requests for busy time information. The message is sent from a "Calendar User" requesting the busy time information to one or more intended recipients.

An "ATTENDEE" property MUST be included for the originator of the request and each of the intended recipients that the method is sent to. The originator is indicated with an "ATTENDEE" property parameter sequence of ";ROLE=ORGANIZER". The recipients are indicated with an "ATTENDEE" property parameter sequence of ";ROLE=ATTENDEE". The requests MAY be fanned out in separate messages to the recipients, with each "REQUEST" method only including the associated "ATTENDEE" properties for the recipients of the message.

If the originator of the "REQUEST" method is not authorized to make a busy time request on the recipient's calendar system, then an exception message is returned in a "REPLY" method, but no busy time data need be returned.

This method type is an iCalendar object that conforms to the following property constraints:

```
     (DESCRIPTION "FREEBUSY - Request For Busy Time" COMPONENT
          "VFREEBUSY" METHOD "REQUEST"

     MUST COMPONENT = CALPROPS (
```



```
          MUST PROPERTY = PRODID, VERSION{"2.0"}, METHOD{"REQUEST"})
     MUST COMPONENT = VFREEBUSY (
          MUST PROPERTY = ATTENDEE{Attendee instances for the Owner and
               Organizer if different and the intended recipient of the
               request}, DTSTAMP, DTSTART, DTEND,SEQUENCE{IF NE 0}, UID
          MAY PROPERTY = SEQUENCE{IF EQ 0}
          NOT PROPERTY = COMMENT, CREATED, DURATION, FREEBUSY,
               LAST-MODIFIED, RELATED-TO, URL, REQUEST-STATUS)
     MAY COMPONENT = VTIMEZONE (
          MUST PROPERTY = DTSTART, TZOFFSET
          MAY PROPERTY = COMMENT, DAYLIGHT, (RDATE / RRULE), TZNAME
          NOT PROPERTY = CREATED)
     MAY COMPONENT X-TOKENS (ANY)
     NOT COMPONENT = VEVENT
     NOT COMPONENT = VTODO
     NOT COMPONENT = VJOURNAL
     NOT COMPONENT = VALARM
     )
```

3.2.3 REPLY

The "REPLY" method in a "VFREEBUSY" calendar component is used to respond to an existing busy time request. The method is sent from a recipient of a busy time request back to the originator of the request. The originator of the request is specified by the "ATTENDEE" property parameter sequence of ";ROLE=ORGANIZER".

The "REPLY" method MAY also be used to respond to an unsuccessful "REQUEST" method. Depending on the "REQUEST-STATUS" value, no busy time information MAY be returned.

Busy time intervals are represented by individual instances of the "FREEBUSY" property. There is one occurrence of the property for each busy time interval. Duplicate busy time periods SHOULD NOT be returned. However, two different busy time periods MAY overlap.

The "FREEBUSY" property value MAY include a list of values, separated by the COMA character (ASCII decimal 44).

"FREEBUSY" properties SHOULD be sorted such that their values are in ascending order, from the most recent to past. For example, today's busy time information SHOULD appear after yesterday's busy time information. And the busy time for this half hour SHOULD appear after the busy time for earlier today.

Since events MAY span a day boundary, free busy time period MAY also span a day boundary.

The busy time periods 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 "ATTENDEE", "DTSTART" and "DTEND"

properties in order to identify the source and date and time range for the busy time data.

The "ATTENDEE" property MUST be specified in the busy time reply. The value is the fully qualified RFC 822 address of the recipient replying to the busy time request.

This method type is an iCalendar object that conforms to the following property constraints:

```
(DESCRIPTION "FreeBusy - Busy Time Reply" COMPONENT "VFREEBUSY"
METHOD "REPLY"

MUST COMPONENT = CALPROPS (
     MUST PROPERTY = PRODID, VERSION{"2.0"}, METHOD{"REPLY"})
MUST COMPONENT = VFREEBUSY (
     MUST PROPERTY = ATTENDEE{address of recipient replying},
          DTSTAMP, DTSTART, DTEND, FREEBUSY{values MUST all be of
          the same data type. Multiple instances are allowed.
          Multiple instances MUST be sorted in ascending order.
          Values MAY NOT overlap}, RELATED-TO{refers to another
          related VFREEBUSY component}, REQUEST-STATUS, UID
     MAY PROPERTY = COMMENT{comment from attendee to originator of
          request}, CREATED{specifies when the busy time data was
          created}, DTSTART{represents start of interval for busy
          time data}, DTEND{represents end of interval for busy
          time data},LAST-MODIFIED{specifies when busy time data
          was last modified}, SEQUENCE{IF EQ 0}, URL{specifies busy
```

```
              time URL}
          NOT PROPERTY = DURATION, SEQUENCE)
     MAY COMPONENT = VTIMEZONE (
          MUST PROPERTY = DTSTART, TZOFFSET
          MAY PROPERTY = COMMENT, DAYLIGHT, (RDATE / RRULE), TZNAME
          NOT PROPERTY CREATED)
     MAY COMPONENT = X-TOKENS (ANY)
     NOT COMPONENT = VEVENT
     NOT COMPONENT = VTODO
     NOT COMPONENT = VJOURNAL
     NOT COMPONENT = VALARM
     )
```

3.3 Methods For VTODO Component

This section defines the property set for the methods that are applicable to the "VTODO" calendar component . Each of the methods is defined using a grammar that specifies the property constraints that define the particular method.

The following summarizes the methods that are defined for the "VTODO" calendar component .

+================+===+

Method	Description
PUBLISH	Post notification of a VTODO. Used primarily as a method of advertising the existence of a VTODO.
REQUEST	Assign a VTODO. This is an explicit assignment to one or more Calendar Users.VTODO REQUEST method is also used to update or change an existing VTODO. Clients that cannot handle REQUEST MAY degrade the method to view it as a PUBLISH.
REPLY	Reply to a VTODO request. Attendees MAYset status to ACCEPTED, DECLINED, TENTATIVE, DELEGATED, PARTIAL, and COMPLETED.
CANCEL	Cancel an existing VTODO assignment.
REFRESH	A request sent to a VTODO "Organizer" asking for the latest version of a
COUNTER	Counter a REQUEST with an alternative proposal.
DECLINECOUNTER	Decline a counter proposal by an attendee.

+================+===+

3.3.1 PUBLISH

The "PUBLISH" method in a "VTODO" calendar component has no reply response associated with it. Instead, it is simply a posting of an iCalendar object that MAY be added to a calendar by a "Calendar User". It requires and accepts no responses to the "Organizer". It's expected usage is for encapsulating an arbitrary "VTODO" calendar component as an iCalendar object. The "Organizer" MAY subsequently update (with another "PUBLISH" method) or cancel (with a "CANCEL" method) a previously published "VTODO" calendar component .

This method type is an iCalendar object that conforms to the following property constraints:

(DESCRIPTION "VTODO - Publish" COMPONENT "VTODO" METHOD "PUBLISH

MUST COMPONENT = CALPROPS (
MUST PROPERTY = PRODID, VERSION{"2.0"}, METHOD{"PUBLISH"})
MUST COMPONENT = VTODO(
MUST PROPERTY = ATTENDEE{only Owner and Organizer if different}, DESCRIPTION{MAYBE NULL}, DTSTAMP, DTSTART, PRIORITY, SEQUENCE{IF NE 0}, UID
MAY PROPERTY = ATTACH, ATTENDEE{other than ROLE=OWNER | ORGANIZER}, CATEGORIES, CLASS, COMMENT, CREATED, DUE, EXDATE, EXRULE, GEO, LAST-MODIFIED, LOCATION,PERCENT-COMPLETE, RELATED-TO, RDATE, RECURRENCE-ID, RESOURCES, RRULE, SEQUENCE{IF EQ 0}, STATUS{TENTATIVE/CONFIRMED /CANCELLED}, SUMMARY{MAYBE NULL}, URL
NOT PROPERTY = REQUEST-STATUS)
MAY COMPONENT = VTIMEZONE (
MUST PROPERTY = DTSTART, TZOFFSET
MAY PROPERTY = COMMENT, DAYLIGHT, (RDATE / RRULE), TZNAME
NOT PROPERTY = CREATED)

MAY COMPONENT = VALARM (
MUST PROPERTY = CATEGORIES, DSTART, DURATION, REPEAT
MAY PROPERTY = ATTACH, DESCRIPTION, SUMMARY
NOT PROPERTY = COMMENT, CREATED, LAST-MODIFIED, RELATED-TO, URL)
MAY COMPONENT = X-TOKENS (ANY)
NOT COMPONENT = VEVENT

NOT COMPONENT = VJOURNAL
NOT COMPONENT = VFREEBUSY
)

3.3.2 REQUEST

The "REQUEST" method in a "VTODO" calendar component provides the following scheduling functions:

- Assign a to-do to one or more "Calendar Users";
- Reschedule an existing to-do;
- Update the details of an existing to-do, without rescheduling it;
- Update the completion status of "Attendees" of an existing to-do, without rescheduling it;
- Reconfirm an existing to-do, without rescheduling it;
- Delegate/reassign an existing to-do to another "Calendar User".

The assigned "Calendar Users" are identified in the "VTODO" calendar component by individual "ATTENDEE;ROLE=ATTENDEE" property value sequences.

The originator of a "REQUEST" is the "Organizer" of the to-do. The recipient of a "REQUEST" is the "Calendar User" assigned the to-do, called the Attendee. The "Attendee" uses the "REPLY" method to convey their acceptance and completion status to the "Organizer" of the "REQUEST".

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 to-do. If the "UID" property value is found on the recipient's calendar, then the "REQUEST" is for either a rescheduling, an update, or a reconfirm of the "VTODO" calendar object.

If the "Organizer" of the "REQUEST" method is not authorized to make a to-do request on the "Attendee's" calendar system, then an exception is returned in the "REQUEST-STATUS" property of a subsequent "REPLY" method, but no scheduling action is performed.

This method type is an iCalendar object that conforms to the following property constraints:

```
(DESCRIPTION "VTODO - Request" COMPONENT "VTODO" METHOD "REQUEST"

MUST COMPONENT = CALPROPS (
     MUST PROPERTY = PRODID, VERSION{"2.0"}, METHOD{"REQUEST"})
MUST COMPONENT = VTODO(
     MUST PROPERTY = ATTENDEE{For Owner and Organizer if different
          and each Attendee},DESCRIPTION{MAYBE NULL}, DTSTAMP,
          DTSTART, PRIORITY, SEQUENCE{IF NE 0}, UID
     MAY PROPERTY = ATTACH, CATEGORIES, CLASS, COMMENT,
          CREATED, DUE, EXDATE, EXRULE, GEO, LAST-MODIFIED,
          LOCATION, PERCENT-COMPLETE, RELATED-TO, RDATE,
          RECURRENCE-ID, RESOURCES, RRULE, SEQUENCE{IF EQ 0},
          STATUS{TENTATIVE/CONFIRMED/CANCELLED}, SUMMARY{MAYBE
          NULL}, URL
     NOT PROPERTY = REQUEST-STATUS)
MAY COMPONENT = VTIMEZONE (
     MUST PROPERTY = DTSTART, TZOFFSET
     MAY PROPERTY = COMMENT, DAYLIGHT, (RDATE / RRULE), TZNAME
     NOT PROPERTY = CREATED)
MAY COMPONENT = VALARM (
     MUST PROPERTY = CATEGORIES, DSTART, DURATION, REPEAT
     MAY PROPERTY = ATTACH, DESCRIPTION, SUMMARY
     NOT PROPERTY =  COMMENT, CREATED, LAST-MODIFIED, RELATED-TO,
          URL)
MAY COMPONENT = X-TOKENS (ANY)
NOT COMPONENT = VEVENT
NOT COMPONENT = VJOURNAL
NOT COMPONENT = VFREEBUSY
)
```

3.3.2.1 REQUEST for Rescheduling a VTODO

The "REQUEST" method MAY be used to reschedule a "VTODO" calendar component .

A rescheduled "VTODO" calendar component involves a change to the existing "VTODO" calendar component in terms of it's start or due time or recurrence intervals and possibly the description. If the recipient "CUA" of a "REQUEST" method finds that the "UID" property value already exists on the calendar, but that the "SEQUENCE" property value in the "REQUEST" is greater than the value for the existing VTODO, then the "REQUEST" method describes a rescheduling of

the "VTODO" calendar component.

3.3.2.2 REQUEST for Update or Reconfirmation of a VTODO

The "REQUEST" method MAY be used to update or reconfirm a "VTODO" calendar component. Reconfirmation is merely an update of "Attendee" completion status or overall "VTODO" calendar component status.

An update to an existing "VTODO" calendar component does not involve changes to the start or due time or recurrence intervals, nor generally to the description for the "VTODO" calendar component. If the recipient "CUA" of a "REQUEST" method finds that the "UID" property value already exists on the calendar and that the "SEQUENCE" property value in the "REQUEST" is the same as the value for the existing event, then the "REQUEST" method describes an update of the "VTODO" calendar component details, but no rescheduling of the "VTODO" calendar component.

The update "REQUEST" is the appropriate response to a "REFRESH" method sent from an "Attendee" to the "Organizer" of a "VTODO" calendar component.

Unsolicited "REQUEST" methods MAY also be sent by the "Organizer" of a "VTODO" calendar component. The unsolicited "REQUEST" methods MAY be used to either update the details of the VTODO, without rescheduling it or to update the completion status of "Attendees" or the "VTODO" calendar component itself (i.e., reconfirm the VTODO).

3.3.2.3 REQUEST for Delegating a VTODO

The "REQUEST" method MAY be used to delegate or reassign ownership of a "VTODO" calendar component to another "Calendar User". The "REQUEST" method is used to delegate the "Attendee's" role (i.e. " "Organizer", or "Attendee") for a "VTODO" calendar component. The "REQUEST" method is sent by one of the "Attendees" of an existing "VTODO" calendar component request to some other individual. In order to avoid scheduling loops, the method MUST NOT be sent from an

"Attendee" back to the "Organizer" of the "VTODO" calendar component. An "Attendee" of a "VTODO" calendar component MAY NOT delegate to the "Organizer" of the event.

For the purposes of this description, the "Attendee" delegating the "VTODO" calendar component is referred to as the "delegator". The "Attendee" receiving the delegation request is referred to as the "delegatee".

The "delegator" of a "VTODO" calendar component MUST forward the existing "REQUEST" method for a "VTODO" calendar component to the "delegatee". The "VTODO" calendar component description MUST include the "delegator's" up-to-date "VTODO" calendar component definition. The "REQUEST" method MUST also include an "ATTENDEE" property with the calendar address of the "delegatee". The "delegator" MUST also send a "REPLY" method back to the "Organizer" with the "delegator's" "Attendee" property "STATUS" parameter value set to DELEGATED. In

addition, the "DELEGATED-TO" parameter SHOULD be included with the calendar address of the "delegatee". A response to the delegation "REQUEST" is sent from the "delegatee" to the "Organizer" and optionally, to the "delegator". The "REPLY" method from the "delegatee" SHOULD include the "ATTENDEE" property with their calendar address and the "DELEGATED-FROM" parameter with the value of the "delegator's"calendar address.

The delegation "REQUEST" method MUST assign the values of the "RSVP" and "EXPECT" property parameters associated with the "delegator's" "Attendee" property to that of the "delegatee's" "Attendee" property. For example if the "delegator's" "Attendee" property specifies "RSVP=TRUE;EXPECT=REQUEST", then the "delegatee's" "Attendee" property MUST specify "RSVP=TRUE;EXPECT=REQUEST".

3.3.2.4 REQUEST Forwarded To An Uninvited Calendar User

An "Attendee" assigned a "VTODO" calendar component MAY also assign the "VTODO" calendar component to another new "Calendar User", not previously associated with the "VTODO" calendar component. The current "Attendee" assigned the "VTODO" calendar component accomplishes this scheduling action by forwarding the original "REQUEST" method to the new "Calendar User".

An "Organizer" of a "VTODO" calendar component MAY also request an

updated completion status from one of the "Attendees". This is achieved by the "Organizer" sending a "REQUEST" method to the "Attendee" with the "ATTENDEE;RSVP=TRUE" property sequence. The "SEQUENCE" property for the "VTODO" calendar component is not changed from its previous value. A recipient will determine that the only change in the "REQUEST" is that their "RSVP" property parameter indicates a request for an updated status. The recipient SHOULD respond with a "REPLY" method indicating their current status with respect to the "REQUEST".

3.3.2.5 REQUEST Updated Attendee Status

An "Organizer" of a to-do MAY also request an updated status from one of the "Attendees". This is achieved by the "Organizer" sending a "REQUEST" method to the "Attendee" with the "ATTENDEE;RSVP=TRUE" property sequence. The "SEQUENCE" property for the to-do is not changed from its previous value. A recipient will determine that the only change in the "REQUEST" is that their "RSVP" property parameter indicates a request for an updated status. The recipient SHOULD respond with a "REPLY" method indicating their current status with respect to the "REQUEST".

This capability MAY also be achieved by the "Organizer" sending the "REFRESH" method to the "Attendees".

3.3.3 REPLY

The "REPLY" method in a "VTODO" calendar component is used to respond (e.g., accept or decline) to a request or to reply to a delegation request. It is also used by an "Attendee" to update their completion status. When used to provide a delegation response, the "delegator" SHOULD include the calendar address of the "delegatee" in the "DELEGATED-TO" parameter of the "delegator's" "ATTENDEE" property. The "delegatee" SHOULD include the calendar address of the "delegator" on the "DELEGATED-FROM" parameter of the "delegatee's" "ATTENDEE" property.

The "REPLY" method MAY also be used to respond to an unsuccessful "VTODO" calendar component "REQUEST" method. Depending on the

"REQUEST-STATUS" value, no scheduling action MAY have been performed.

The "Organizer" of a "VTODO" calendar component MAY receive a "REPLY" method from a "Calendar User" not in the original "REQUEST". For example, a "REPLY" method MAY be received from a "delegatee" of a "VTODO" calendar component. In addition, the "REPLY" method MAY be received from an unknown "Calendar User"; forwarded the "REQUEST" from an original "Attendee" assigned the "VTODO" calendar component. This uninvited "Attendee" MAY be accepted, or the "Organizer" MAY cancel the "VTODO" calendar component for the uninvited "Attendee" by sending them a "CANCEL" method.

This method type is an iCalendar object that conforms to the following property constraints:

```
  (DESCRIPTION "VTODO - Reply" COMPONENT "VTODO" METHOD "REPLY"

  MUST COMPONENT = CALPROPS (
       MUST PROPERTY = PRODID, VERSION{"2.0"}, METHOD{"REPLY"})
  MUST COMPONENT = VTODO(
       MUST PROPERTY = ATTENDEE{MUST be address of ATTENDEE
            replying}, DTSTAMP, SEQUENCE{IF NE 0}, UID{UID
            associated with original REQUEST}
       MAY PROPERTY = COMMENT{provides comment from ATTENDEE to
            "Organizer"}, EXDATE, EXRULE, RECURRENCE-ID, REQUEST-
            STATUS, PERCENT-COMPLETE, SEQUENCE{IF EQ 0}, SUMMARY
            {MAYBE NULL}, URL
       NOT PROPERTY = ATTACH, CATEGORIES, CLASS, CREATED,
            DESCRIPTION, DTSTART, DUE, GEO, LAST-MODIFIED, PRIORITY,
            RELATED-TO, RESOURCES, RDATE, RRULE, STATUS, SUMMARY,
            TRANSP, URL)
  MAY COMPONENT = X-TOKENS (ANY)
  NOT COMPONENT = VEVENT
  NOT COMPONENT = VJOURNAL
  NOT COMPONENT = VFREEBUSY
  NOT COMPONENT = VTIMEZONE
  NOT COMPONENT = VALARM
  )
```


3.3.4 CANCEL

The "CANCEL" method in a "VTODO" calendar component is used to send a

cancellation notice of an existing "VTODO" calendar request request to the "Attendees". The message is sent by the "Organizer" of a "VTODO" calendar component to the "Attendees" of the "VTODO" calendar component. For a recurring "VTODO" calendar component, either the whole "VTODO" calendar component or instances of a "VTODO" calendar component MAY be cancelled. To cancel the complete range of a recurring "VTODO" calendar component, the "UID" property value for the "VTODO" calendar component MUST be specified in the "CANCEL" method. In order to cancel an individual instance of a recurring "VTODO" calendar component, the "RECURRENCE-ID" property value for the "VTODO" calendar component MUST be specified in the "CANCEL" method. In order to cancel a sequence of instances of a recurring "VTODO" calendar component, either the "RECURRENCE-ID" property for the first instance in the sequence MUST be specified with the "RANGE" property parameter value of THISANDPRIOR or THISANDFUTURE; to indicate cancellation of the "VTODO" calendar component instances before and after the first instance, respectively. Lastly, individual recurrence instances MAY be cancelled by specifying multiple "RECURRENCE-ID" properties corresponding to the instances to be cancelled.

This method type is an iCalendar object that conforms to the following property constraints:

```
  (DESCRIPTION "VTODO - Cancel" COMPONENT "VTODO" METHOD "CANCEL"

  MUST COMPONENT = CALPROPS (
       MUST PROPERTY = PRODID, VERSION{"2.0"}, METHOD{"CANCEL"})
  MUST COMPONENT = VTODO(
       MUST PROPERTY = DTSTAMP, SEQUENCE{IF NE 0}, UID{UID
            associated with original REQUEST}
       MAY PROPERTY = COMMENT{provides comment from "Organizer" to
            ATTENDEE}, EXDATE, EXRULE, RECURRENCE-ID, SEQUENCE{IF EQ
            0}, STATUS{CANCELLED}
       NOT PROPERTY = ATTACH, ATTENDEE, CATEGORIES, CLASS, CREATED,
            DESCRIPTION, DTSTART, DUE, GEO, LAST-MODIFIED, PRIORITY,
            RDATE, RELATED-TO, RESOURCES, REQUEST-STATUS, RRULE,
            SUMMARY, TRANSP, URL)
  MAY COMPONENT = X-TOKENS (ANY)
  NOT COMPONENT = VEVENT
  NOT COMPONENT = VJOURNAL
  NOT COMPONENT = VFREEBUSY
  NOT COMPONENT = VTIMEZONE
  NOT COMPONENT = VALARM
  )
```

3.3.5 REFRESH

The "REFRESH" method in a "VTODO" calendar component is used by "Attendees" of an existing "VTODO" calendar component to request an updated description from the "Organizer" of the "VTODO" calendar component. The "Organizer" of the "VTODO" calendar component MAY also use this method to request an updated status from the "Attendees". The "REFRESH" method MUST specify the "UID" property corresponding to the "VTODO" calendar component needing update.

A refresh of a recurrence instance of a "VTODO" calendar component MAY be requested by specifying the "RECURRENCE-ID" property corresponding to the associated "VTODO" calendar component. The "Organizer" MUST respond with the latest description and rendition of the "VTODO" calendar component. This method is intended to facilitate machine processing of requests for updates to a "VTODO" calendar component.

This method type is an iCalendar object that conforms to the following property constraints:

```
(DESCRIPTION "VTODO - Refresh" COMPONENT "VTODO" METHOD "REFRESH"

MUST COMPONENT = CALPROPS (
     MUST PROPERTY = PRODID, VERSION{"2.0"}, METHOD{"REFRESH"})
MUST COMPONENT = VTODO(
     MUST PROPERTY = ATTENDEE{address of requestor}, DTSTAMP,
          SEQUENCE{IF NE 0}, UID{UID associated with original
          REQUEST} MAY PROPERTY = COMMENT{provides comment from
          ATTENDEE to "Organizer"}, RECURRENCE-ID, SEQUENCE{IF EQ
          0}
     NOT PROPERTY = ATTACH, CATEGORIES, CLASS, CREATED,
          DESCRIPTION, DTSTART, DUE, EXDATE, EXRULE, GEO, LAST-
          MODIFIED, PRIORITY, RDATE,RELATED-TO, RESOURCES,
          REQUEST-STATUS, RRULE,SUMMARY, STATUS, TRANSP, URL)
MAY COMPONENT = X-TOKENS (ANY)
NOT COMPONENT = VEVENT
NOT COMPONENT = VJOURNAL
NOT COMPONENT = VFREEBUSY
NOT COMPONENT = VTIMEZONE
NOT COMPONENT = VALARM
)
```

3.3.6 COUNTER

The "COUNTER" method in a "VTODO" calendar component is used by an "Attendee" of an existing "VTODO" calendar component to submit to the "Organizer" a counter proposal for the "VTODO" calendar component. The "Attendee" MUST send the message to the "Organizer" of the "VTODO" calendar component.

The counter proposal is an iCalendar object consisting of a "VTODO" calendar component describing the complete description of the alternate "VTODO" calendar component.

The "Organizer" rejects the counter proposal by sending the "Attendee" a "DECLINE-COUNTER" method. The "Organizer" accepts the counter proposal by sending all of the "Attendees" of the "VTODO" calendar component a "REQUEST" method rescheduling the "VTODO" calendar component. In the later case, the "Organizer" SHOULD reset the individual "RSVP" property parameter values to TRUE on each "ATTENDEE" property; in order to force a response by the "Attendees".

This method type is an iCalendar object that conforms to the following property constraints:

```
  (DESCRIPTION "VTODO- Request" COMPONENT "VTODO" METHOD "REQUEST"

  MUST COMPONENT = CALPROPS (
       MUST PROPERTY = PRODID, VERSION{"2.0"}, METHOD{"REQUEST"})
  MUST COMPONENT = VTODO(
       MUST PROPERTY = ATTENDEE{For Owner and Orginator if different
            and Attendees}, DESCRIPTION{MAYBE NULL}, DTSTAMP,
            DTSTART, PRIORITY, SEQUENCE{IF NE 0}, UID
       MAY PROPERTY = ATTACH, CATEGORIES, CLASS, COMMENT,CREATED,
            DUE, EXDATE, EXRULE, GEO, LAST-MODIFIED,
            LOCATION,RELATED-TO, RDATE, RECURRENCE-ID, RESOURCES,
            RRULE, SEQUENCE{IF EQ 0}, STATUS{TENTATIVE/CONFIRMED
            /CANCELLED}, SUMMARY{MAYBE NULL}, URL
       NOT PROPERTY = REQUEST-STATUS)
  MAY COMPONENT = VTIMEZONE (
       MUST PROPERTY = DTSTART, TZOFFSET
       MAY PROPERTY = COMMENT, DAYLIGHT, (RDATE / RRULE), TZNAME
       NOT PROPERTY = CREATED)
  MAY COMPONENT = VALARM (
```

```
          MUST PROPERTY = CATEGORIES, DSTART, DURATION, REPEAT
          MAY PROPERTY = ATTACH, DESCRIPTION, SUMMARY
       NOT PROPERTY =  COMMENT, CREATED, LAST-MODIFIED, RELATED-TO, URL)
       MAY COMPONENT = X-TOKENS (ANY)
       NOT COMPONENT = VEVENT
       NOT COMPONENT = VJOURNAL
       NOT COMPONENT = VFREEBUSY
       )
```

3.3.7 DECLINECOUNTER

The "DECLINE-COUNTER" method in a "VTODO" calendar component is used by an "Organizer" of "VTODO" calendar component to reject a counter proposal offered by one of the "Attendees". The "Organizer" MUST send the message to the "Attendee" that sent the "COUNTER" method to the "Organizer".

The counter proposal is an iCalendar object consisting of a "VTODO" calendar component describing the complete description of the alternate "VTODO" calendar component.

The "Organizer" rejects the counter proposal by sending the "Attendee" a "DECLINE-COUNTER" method. The "Organizer" accepts the counter proposal by sending all of the "Attendees" of the "VTODO" calendar component a "REQUEST" method rescheduling the "VTODO" calendar component. Since this is a rescheduled "VTODO", the "SEQUENCE" property value will be incremented. In the later case, the "Organizer" SHOULD reset the individual "RSVP" property parameter values to TRUE on all of the "ATTENDEE" properties in order to force a response by the "Attendees".

This method type is an iCalendar object that conforms to the following property constraints:

```
  (DESCRIPTION "VTODO - Cancel" COMPONENT "VTODO" METHOD
  "DECLINECOUNTER"

  MUST COMPONENT = CALPROPS (
       MUST PROPERTY = PRODID, VERSION{"2.0"}, METHOD
            {"DECLINECOUNTER"})
```

```
     MUST COMPONENT = VTODO(
          MUST PROPERTY = DTSTAMP, SEQUENCE{IF NE 0}, UID{same UID
               specified In Original REQUEST and subsequent COUNTER}
          MAY PROPERTY = COMMENT{provides comment from "Organizer" to
               ATTENDEE}, RECURRENCE-ID, REQUEST-STATUS, SEQUENCE{IF EQ
               0}
          NOT PROPERTY = ATTACH, ATTENDEE, CATEGORIES, CLASS, CREATED,
               DESCRIPTION, DTSTART, DUE, EXDATE, EXRULE, GEO, LAST-
               MODIFIED, PRIORITY, RDATE, RELATED-TO, RESOURCES, RRULE,
               STATUS, SUMMARY, TRANSP, URL)
     MAY COMPONENT = X-TOKENS (ANY)
     NOT COMPONENT = VEVENT
     NOT COMPONENT = VJOURNAL
     NOT COMPONENT = VFREEBUSY
     NOT COMPONENT = VTIMEZONE
     NOT COMPONENT = VALARM
     )
```

3.4 Methods For VJOURNAL Component

This section defines the property set for the methods that are applicable to the "VJOURNAL" calendar component. Each of the methods is defined using a grammar that clarifies the property constraints that define the particular method.

The following summarizes the methods that are defined for the "VJOURNAL" calendar component.

Method	Description

PUBLISH	Post a journal entry. Used primarily as a method of advertising the existence of a journal entry
CANCEL	Cancel an existing journal entry request.
REFRESH	A request sent to the journal "Organizer" for the latest version of the journal entry to be resent the requester.

3.4.1 PUBLISH

The "PUBLISH" method in a "VJOURNAL" calendar component has no reply response associated with it. Instead, it is simply a posting of an iCalendar object that MAY be added to a calendar by a "Calendar User" agent. It requires and accepts no responses to the "Organizer". The expected usage is for encapsulating an arbitrary journal entry as an iCalendar object. The "Organizer" MAY subsequently update (with another "PUBLISH" method) or cancel (with a "CANCEL" method) a previously published journal entry.

This method type is an iCalendar object that conforms to the following property constraints:

```
  (DESCRIPTION "Journal - Publish" COMPONENT "VJOURNAL" METHOD
       "PUBLISH

  MUST COMPONENT = CALPROPS (
       MUST PROPERTY = PRODID, VERSION{"2.0"}, METHOD{"PUBLISH"})
  MUST COMPONENT = VJOURNAL (
       MUST PROPERTY = DESCRIPTION{MAYBE NULL}, DTSTAMP,
            DTSTART{VALUE=DATE}, SEQUENCE{IF NE 0}, UID
       MAY PROPERTY = ATTACH, ATTENDEE{only ROLE=ORGANIZER and
            OWNER if different}, CATEGORIES, CLASS, COMMENT,
            CREATED, EXDATE, EXRULE, LAST-MODIFIED, RELATED-TO,
            RDATE, RECURRENCE-ID, RRULE, SEQUENCE{IF EQ 0},
            SUMMARY{MAYBE NULL}, URL
       NOT PROPERTY = REQUEST-STATUS)
  MAY COMPONENT = VTIMEZONE (
       MUST PROPERTY = DTSTART, TZOFFSET
       MAY PROPERTY = COMMENT, DAYLIGHT, (RDATE / RRULE), TZNAME
       NOT PROPERTY = CREATED)
  MAY COMPONENT = X-TOKENS (ANY)
  NOT COMPONENT = VEVENT
  NOT COMPONENT = VTODO
  NOT COMPONENT = VALARM
  NOT COMPONENT = VFREEBUSY
  )
```


3.4.2 CANCEL

The "CANCEL" method in a "VJOURNAL" calendar component is used to send a cancellation notice of an existing journal entry request to the "Attendees". The message is sent by the "Organizer" of a journal entry to the "Attendees" of the journal entry. For a recurring journal entry, either the whole journal entry or instances of a journal entry MAY be cancelled. To cancel the complete range of a recurring journal entry, the "UID" property value for the journal entry MUST be specified in the "CANCEL" method. In order to cancel an individual instance of the journal entry, the "RECURRENCE-ID" property value for the journal entry MUST be specified in the "CANCEL" method. In order to cancel a sequence of instances in a recurring journal entry, the "RECURRENCE-ID" property for the first journal entry instance in the sequence MUST be specified with the "RANGE" property parameter value of THISANDPRIOR or THISANDFUTURE; to indicate cancellation of the journal entry instances before and after the first journal entry instance, respectively. Lastly, individual recurrence instances MAY be cancelled by specifying multiple "RECURRENCE-ID" properties corresponding to the instances to be cancelled.

This method type is an iCalendar object that conforms to the following property constraints:

```
  (DESCRIPTION "Journal - Cancel" COMPONENT "VJOURNAL" METHOD
       "CANCEL"

  MUST COMPONENT = CALPROPS (
       MUST PROPERTY = PRODID, VERSION{"2.0"}, METHOD{"CANCEL"})
  MUST COMPONENT = VJOURNAL (
       MUST PROPERTY = DTSTAMP, SEQUENCE{IF NE 0}, UID{UID
            associated with original REQUEST}
       MAY PROPERTY = COMMENT{provides comment from "Organizer" to
            ATTENDEE}, EXDATE, EXRULE, RECURRENCE-ID, SEQUENCE{IF EQ
            0}, STATUS{CANCELLED}
       NOT PROPERTY = ATTACH, ATTENDEE, CATEGORIES, CLASS, CREATED,
            DESCRIPTION, DTSTART, LAST-MODIFIED, RDATE, RELATED-TO,
            REQUEST-STATUS, RRULE, SUMMARY, URL)
  MAY COMPONENT = X-TOKENS (ANY)
  NOT COMPONENT = VEVENT
  NOT COMPONENT = VTODO
  NOT COMPONENT = VFREEBUSY
  NOT COMPONENT = VTIMEZONE
  NOT COMPONENT = VALARM
  )
```

3.4.3 REFRESH

The "REFRESH" method in a "VJOURNAL" calendar component is used by

"Attendees" of an existing journal entry to request an updated description from the journal entry "Organizer". The "REFRESH" method

MUST specify the "UID" property corresponding to the journal entry needing update. A recurrence instance of a journal entry MAY be requested by specifying the"RECURRENCE-ID" property corresponding to the associated journal entry. The "Organizer" MUST respond with the latest description and version of the journal entry. This method is intended to facilitate machine processing of the "REFRESH" response.

This method type is an iCalendar object that conforms to the following property constraints:

```
  (DESCRIPTION "Journal - Refresh" COMPONENT "VJOURNAL" METHOD
       "REFRESH"

  MUST COMPONENT = CALPROPS (
       MUST PROPERTY = PRODID, VERSION{"2.0"}, METHOD{"REFRESH"})
  MUST COMPONENT = VJOURNAL (
       MUST PROPERTY = ATTENDEE{address of requestor}, DTSTAMP,
            SEQUENCE{IF NE 0}, UID{UID associated with original
            REQUEST}
       MAY PROPERTY = COMMENT{provides comment from ATTENDEE to
            "Organizer"}, RECURRENCE-ID, SEQUENCE{IF EQ 0}
       NOT PROPERTY = ATTACH, CATEGORIES, CLASS, CREATED,
            DESCRIPTION, DTSTART, EXDATE, EXRULE, LAST-MODIFIED,
            PRIORITY, RDATE, RELATED-TO, REQUEST-STATUS, RRULE,
            SUMMARY, URL)
  MAY COMPONENT = X-TOKENS (ANY)
  NOT COMPONENT = VEVENT
  NOT COMPONENT = VTODO
  NOT COMPONENT = VFREEBUSY
  NOT COMPONENT = VTIMEZONE
  NOT COMPONENT = VALARM
  )
```

3.5 Status Replies

The "REQUEST-STATUS" property MAY include the following values:

Short Return	Longer Return Status	Offending Data

Status Code	Description	
2.0	Success.	None.
2.1	Success but fallback taken on one or more property values.	Property name and value MAY be specified.
2.2	Success, invalid property ignored.	Property name MAY be specified.
2.3	Success, invalid property parameter ignored.	Property parameter name and value MAY be

		specified.
2.4	Success, unknown non-standard property ignored.	Non-standard property name MAY be specified.
2.5	Success, unknown non standard property value ignored.	Property and non-standard value MAY be specified.
2.6	Success, invalid calendar component ignored.	Calendar component sentinel (e.g., "BEGIN: ALARM") MAY be specified.
2.7	Success, request forwarded to Calendar User.	Original and forwarded caluser addresses MAY be specified.
2.8	Success, repeating event ignored. Scheduled as a single event.	RRULE or RDATE property name and value MAY be specified.
2.9	Success, truncated end date time to date boundary.	DTEND property value MAY be specified.
2.10	Success, repeating VTODO ignored. Scheduled as a single VTODO.	RRULE or RDATE property name and value MAY be specified.

3.0	Invalid property name.	Property name MAY be specified.
3.1	Invalid property value.	Property name and value MAY be specified.
3.2	Invalid property parameter.	Property parameter name and value MAY be specified.
3.3	Invalid property parameter value.	Property parameter name and value MAY be specified.
3.4	Invalid calendar component sequence.	Calendar component sentinel MAY be specified (e.g., BEGIN: VTIMEZONE).
3.5	Invalid date or time.	Date/time value(s) MAY be specified.
3.6	Invalid rule.	Rule value MAY be specified.

3.7	Invalid Calendar User.	Attendee property value MAY be specified.
3.8	No authority.	PROFILE and ATTENDEE property values MAY be specified.
3.9	Unsupported version.	VERSION property name and value MAY be specified.
3.10	Request entity too large.	None.
4.0	Event conflict. Date/time is busy.	DTSTART and DTEND property name and values MAY be specified.

5.0	Request not supported.	Method property value MAY be specified.
5.1	Service unavailable.	ATTENDEE property value MAY be specified.
5.2	Invalid calendar service.	ATTENDEE property value MAY be specified.
5.3	No scheduling support for user.	ATTENDEE property value MAY be specified.

3.6 Implementation Considerations

3.6.1 Working With Recurrence Instances

iCalendar includes a recurrence grammar to represent recurring events. The benefit of such a grammar is the ability to represent a number of events in a single object. However, while this simplifies creation of a recurring event, meeting instances MAY still need to be referenced. For instance, an "Attendee" MAY decline the third instance of a recurring Friday event. Similarly, the "Organizer" MAY change the time or location to a single instance of the recurring event.

Since implementations MAY elect to store recurring events as either a single event object or a collection of discreet, related event objects, the protocol is designed so that each recurring instance MAY be both referenced and versioned. Hence, implementations that choose to maintain per-instance properties (such as "ATTENDEE" property "STATUS" parameter) MAY do so. However, the protocol does not require per-instance recognition unless the instance itself MUST be renegotiated.

The scenarios for recurrence instance referencing are listed below. For purposes of simplification a change to an event refers to a "trigger property." That is, a property that has a substantive affect on the meeting itself such as start time, location, due date (for "VTODO" calendar component components) and possibly description.

. "Organizer" initiated actions:

. "Organizer" deletes or changes a single instance of a recurring event

. "Organizer" makes changes that affect all future instances

. "Organizer" makes changes that affect all previous instances

. "Organizer" deletes or modifies a previously changed instance

. Attendee initiated actions:

. Attendee changes status for a particular recurrence instance

. Attendee sends Event-Counter for a particular recurrence instance

An instance of a recurring event is assigned a unique identification, "RECURRENCE-ID" property, when that instance MUST be renegotiated. Negotiation is necessary when the start time, end time, due date or location are modified. If the "Organizer" wishes to identify a specific recurrence instance it is done using the "RECURRENCE-ID" property. The property value is equal to the date/time of the instance. If the "Organizer" wishes to change the"DTSTART", the original "DTSTART" value is used for"RECURRENCE-ID" property and the new "DTSTART" and "DTEND" values reflect the change. If the "Organizer" wishes to add a new instance to the recurring event then a "REQUEST" is issued with an "RDATE" property equal to the new instance date. It is recommended that the "Organizer" include the "RECURRENCE-ID" property[SS1]. Since the creation of a new event instance requires negotiation, the sequence number is also incremented.

3.6.2 Attendee Property Considerations

The "ATTENDEE" property for the "Organizer" is required on published events, to-dos, and journal entries for two reasons. First, a published only the "Organizer" is allowed to update an event, to-do, or journal entry component. The "Organizer" "ATTENDEE" property MUST be present in the event, to-do, or journal entry component so that the "CUA" has a basis for authorizing an update. Second, it is prudent to provide a point of contact for anyone who receives a published component in case of problems.

There are valid RFC 822 addresses that represent groups. Sending email to such an address results in mail being sent to multiple

recipients. Such an address MAY be used as the value of an "ATTENDEE" property. Thus, it is possible that the recipient of a "REQUEST" does not appear explicitly in the list list.

It is recommended that the general approach to finding a "Calendar User" in an attendee list be as follows:

1. Search for the "Calendar User" in the attendee list where "TYPE=INDIVIDUAL"

2. Failing (1) look for attendees where "TYPE=GROUP" or 'TYPE=UNKNOWN". The "CUA" MUST then determine if the "CU" is a member of one of these groups. If so, the "REPLY" method sent to the "Organizer" MUST contain a new "ATTENDEE" property in which the "TYPE" property parameter is set to INDIVIDUAL and the "GROUP" property parameter is set to the name of the group.

3. Failing (2) the "CUA" MAY ignore or accept the request as the "Calendar User" wishes.

3.6.3 When To Refresh An Event

An "VEVENT" or "VTODO" calendar component SHOULD be resent to all "Attendees" whenever the "SEQUENCE" property is incremented or any other substantive change is made.

3.6.4 Timezones

If a recurring event has any instance where "DTSTART" and "DTEND" fall on different sides of a time zone shift, the "VTIMEZONE" components are required.

The threat of duplicate time zone definitions exists. SHOULD an iCalendar object contain multiple conflicting time zone components, the one with the latest "DTSTART" property supersedes the others.

3.6.5 Alarms

It is recommended that application software ask the user whether or not they want alarms included when they read the event.

3.6.6 SUMMARY Property

The minimum support for the "SUMMARY" property in a recipient MUST be for a 255 byte value. Implementations MAY truncate longer length values.

3.6.7 X-Tokens

To make iCalendar objects extensible, new property types MAY be inserted into components. These properties are called X-Tokens as they are prefixed with "X-". A client is not required to make sense of X-Tokens. Clients are not required to save X-Tokens or use them in event replies.

4 Examples

4.1 Published Event Examples

In the calendaring and scheduling context, publication refers to the one way transfer of event information. Consumers of published events simply incorporate the event into a calendar. No reply is expected. Individual "A" publishes an event. Individual "B" reads the event and incorporates it into their calendar. Events MAY be published in several ways including: embedding the event as an object in a web page, e-mailing the event to a distribution list, and posting the event to a newsgroup.

The table below illustrates the sequence of events between the publisher and the consumers of a published event.

Action	"Organizer"
Publish an event	"A" sends or posts a PUBLISH message
"B" reads a published event	

Publish an updated event	"A" sends or posts a PUBLISH message
"B" reads the updated event	
Cancel a published event	"A" sends or posts a CANCEL message
"B" reads the canceled event publication	

4.1.1 A Minimal Published Event

The iCalendar object below describes a single event that begins on July 1, 1997 at 20:00 UTC. This event contains the minimum set of properties for a "PUBLISH" for a "VEVENT" calendar component.


```
BEGIN:VCALENDAR
METHOD:PUBLISH
PRODID:-//ACME/DesktopCalendar//EN
VERSION:2.0
BEGIN:VEVENT
ATTENDEE;ROLE=OWNER:mailto:a@host.com
DTSTART:19970701T200000Z
DTSTAMP:19970611T190000Z
SUMMARY:ST. PAUL SAINTS -VS- DULUTH-SUPERIOR DUKES
UID:0981234-1234234-23@host.com
END:VEVENT
END:VCALENDAR
```

4.1.2 Changing A Published Event

The iCalendar object below describes an update to the event described in 4.1.1, the time has been changed, an end time has been added, and the sequence number has been adjusted.

```
BEGIN:VCALENDAR
METHOD:PUBLISH
```

```
VERSION:2.0
PRODID:-//ACME/DesktopCalendar//EN
BEGIN:VEVENT
ATTENDEE;ROLE=OWNER:mailto:A@example.com
DTSTAMP:19970612T190000Z
DTSTART:19970701T210000Z
DTEND:19970701T230000Z
SEQUENCE:2
UID:0981234-1234234-23@host.com
SUMMARY:ST. PAUL SAINTS -VS- DULUTH-SUPERIOR DUKES
END:VEVENT
END:VCALENDAR
```

The "UID" property is used by the client to identify the event. The "SEQUENCE" property indicates that this is the second change to the event. Events with sequence numbers 0 and 1 are superseded by this event.

The "SEQUENCE" property provides a reliable way to distinguish different versions of the same event. Each time an event is published, its sequence number is incremented. If a client receives an event with a sequence number 5 and finds it has the same event with sequence number 2, the event SHOULD be updated. However, if the client received an event with sequence number 2 and finds it already has sequence number 5 of the same event, the event SHOULD NOT be updated.

4.1.3 Canceling A Published Event

The iCalendar object below cancels the event described in 4.1.1. This cancels the event with "SEQUENCE" property of 0, 1, and 2.

```
BEGIN:VCALENDAR
METHOD:CANCEL
VERSION:2.0
PRODID:-//ACME/DesktopCalendar//EN
BEGIN:VEVENT
ATTENDEE;ROLE=OWNER:mailto:A@example.com
```

```
COMMENT:DUKES forfeit the game
SEQUENCE:2
UID:0981234-1234234-23@host.com
DTSTAMP:19970613T190000Z
STATUS:CANCELLED
END:VEVENT
END:VCALENDAR
```

4.1.4 A Rich Published Event

This example describes the same event as in 4.1.1, but in much greater detail.

```
BEGIN:VCALENDAR

PRODID:-//ACME/DesktopCalendar//EN
METHOD:PUBLISH
SCALE:GREGORIAN
SOURCE:http://www.midwaystadium.com/stadium-cal/1997-events.or4
NAME:1997 GAME SCHEDULE
VERSION:2.0

BEGIN:VTIMEZONE
DAYLIGHT:TRUE
DTSTART:19970406T070000-0600
RRULE:FREQ=YEARLY;BYDAY=1SU;BYMONTH=4
TZNAME:CDT
TZOFFSET:-0500
END:VTIMEZONE

BEGIN:VTIMEZONE
DAYLIGHT:FALSE
DTSTART:19971026T0200-0500
RRULE:FREQ=YEARLY;BYDAY=-1SU;BYMONTH=10
TZNAME:CST
TZOFFSET:-0600
END:VTIMEZONE

BEGIN:VEVENT
ATTENDEE;ROLE=OWNER:mailto:A@example.com
```



```
ATTACH:http://www.midwaystadium.com
```

```
  CATEGORIES:SPORTS EVENT;ENTERTAINMENT
  CLASS:PRIVATE
  CREATED:19970415T194319Z
  DESCRIPTION:MIDWAY STADIUM\n
   Big time game. MUST see.\n
   Expected duration:2 hours\n
  DTEND:19970701T180000
  DTSTART:19970702T160000
  DTSTAMP:19970614T190000Z
  STATUS:CONFIRMED
  LAST-MODIFIED:19970416T162421Z
  LOCATION;VALUE=URL:http://www.midwaystadium.com/
  PRIORITY:2
  RESOURCES:SCOREBOARD
  SEQUENCE:3
  SUMMARY:ST. PAUL SAINTS -VS- DULUTH-SUPERIOR DUKES
  UID:0981234-1234234-23@host.com
  RELATED-TO:0981234-1234234-14@host.com

  BEGIN:VALARM
  DTSTART:19970701T190000Z
  REPEAT:2
  DURATION:PT2H
  CATEGORIES:DISPLAY,AUDIO
  DESCRIPTION:It's almost game time
  END:VALARM

  BEGIN:VALARM
  DTSTART:19970701T153000
  CATEGORIES:DISPLAY,AUDIO
  DESCRIPTION:You SHOULD leave now. Game starts in 30 min!
  END:VALARM

  END:VEVENT
  END:VCALENDAR
```

The "CLASS" property is specified, though it is not really needed here. Since this is a published event, a program that displays it need not apply any content filtering based on the "CLASS" attribute. If this event is copied into a user's calendar, the "CLASS" would be included as part of the copy. The handling of the "CLASS" tag at that point is implementation specific.

The "RELATED-TO" field contains the "UID" property of a related calendar event. The handling of this property is application dependent.

The "SEQUENCE" property 3 indicates that this event supersedes versions 0, 1, and 2.

4.1.5 Anniversaries or Events attached to entire days

This example demonstrates the use of the "value" parameter to tie a VEVENT to day rather than a specific time.

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:PUBLISH
VERSION:2.0
BEGIN:VEVENT
DTSTAMP:19970614T190000Z
UID:0981234-1234234-23@host.com
DTSTART;VALUE=DATE:19970714
RRULE:FREQ=YEARLY;INTERVAL=1
SUMMARY: Bastille Day
END:VEVENT
END:VCALENDAR
```

4.2 Group Event Examples

Group events are distinguished from published events in that they have "Attendees" and that there is interaction between the "Attendees" with respect to the event. Individual "A" requests a meeting between individuals "A", "B", "C" and "D". Individual "B" confirms attendance to the meeting. Individual "C" declines attendance. Individual "D" tentatively confirms their attendance. This is sometimes referred to as "penciling-in" the event on a calendar. The following table illustrates the sequence of messages that would be exchanged between these individuals. The table below illustrates the message flow.

Action	"Organizer"	Attendee
Initiate a meeting request	"A" sends a REQUEST message to "B", "C", and "D"	

Accept the meeting request		"B" sends a REPLY message to "A" with its ATTENDEE STATUS para- set to "ACCEPTED"
Decline the meeting request		"C" sends a REPLY message to "A" with its ATTENDEE STATUS para- set to "DECLINED"
Tentatively accept the meeting request		"D" sends a REPLY message to "A" with its ATTENDEE STATUS para-

		set to "TENTATIVE"
Confirm meeting status with attendees	"A" sends a REQUEST message to "B" and "C" with updated information.	

4.2.1 A Group Event Request

A sample meeting request that "A" sends to "B", "C", and "D".

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:REQUEST
VERSION:2.0
BEGIN:VEVENT
ATTENDEE;ROLE=OWNER:Mailto:A@example.com
ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:B@example.
com
ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:C@example.
com
ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:D@example.
com
ATTENDEE;RSVP=FALSE;EXPECT=REQUIRE;TYPE=ROOM:CR_Big@example.com
DTSTAMP:19970611T190000Z
DTSTART:19970701T100000-0700
```

```
DTEND:19970701T103000-0700
SUMMARY:Phone Conference
UID:www.acme.com-873970198738777@host.com
SEQUENCE:0
STATUS:CONFIRMED
END:VEVENT
END:VCALENDAR
```

4.2.2 Reply To A Group Event Request

Attendee "B" accepts the meeting.

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:REPLY
VERSION:2.0
BEGIN:VEVENT
ATTENDEE;STATUS=ACCEPTED:Mailto:B@example.com
UID:www.acme.com-873970198738777@host.com
SEQUENCE:0
REQUEST-STATUS:2.0;Success
DTSTAMP:19970612T190000Z
```



```
END:VEVENT
END:VCALENDAR
```

"B" could have declined the meeting or indicated tentative acceptance by setting the ATTENDEE;"STATUS" parameter to DECLINED or TENTATIVE, respectively.

4.2.3 Update An Event

The event is moved to a different time. The combination of the "UID" property(which remains the same) and the SEQUENCE (bumped to 1) properties indicate the update.

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:REQUEST
VERSION:2.0
```

```
     BEGIN:VEVENT
     ATTENDEE;ROLE=OWNER:Mailto:A@example.com
     ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
     ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:B@example.
com
     ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:C@example.
com
     ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:D@example.
com
     ATTENDEE;RSVP=FALSE;EXPECT=REQUIRE;TYPE=ROOM:CR_Big@example.com
     DTSTART:19970701T110000-0700
     DTEND:19970701T113000-0700
     SUMMARY:Phone Conference
     UID:www.acme.com-873970198738777@host.com
     SEQUENCE:1
     DTSTAMP:19970613T190000Z
     STATUS:CONFIRMED
     END:VEVENT
     END:VCALENDAR
```

4.2.4 Countering an Event Proposal

Attendee A sends "REQUEST" to B and C. B makes a counter proposal to A to change the time and location.

Attendee A sends "REQUEST":

```
     BEGIN:VCALENDAR
     PRODID:-//ACME/DesktopCalendar//EN
     METHOD:REQUEST
     VERSION:2.0
     BEGIN:VEVENT
     ATTENDEE;ROLE=OWNER:Mailto:A@example.com
```



```
     ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
     ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:B@example.
     com
     ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:C@example.
com
     DTSTART:19970701T190000Z
     DTEND:19970701T200000Z
     SUMMARY:Discuss the Merits of the election results
```

```
      LOCATION:The Big Conference Room
      UID:www.acme.com-873970198738777@host.com
      SEQUENCE:0
      DTSTAMP:19970611T190000Z
      STATUS:CONFIRMED
      END:VEVENT
      END:VCALENDAR
```

Attendee B sends "COUNTER" to A, requesting changes to time and place:

```
      BEGIN:VCALENDAR
      PRODID:-//ACME/DesktopCalendar//EN
      METHOD:COUNTER
      VERSION:2.0
      BEGIN:VEVENT
      ATTENDEE;ROLE=OWNER:Mailto:A@example.com
      ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
      ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:B@example.
      com
      ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:C@example.
com
      DTSTART:19970701T160000Z
      DTEND:19970701T190000Z
      DTSTAMP:19970612T190000Z
      SUMMARY:Discuss the Merits of the election results
      LOCATION:The Small Conference Room
      COMMENT:This time works much better and I think the big conference
        room is too big
      UID:www.acme.com-873970198738777@host.com
      SEQUENCE:0
      DTSTAMP:19970611T190000Z
      END:VEVENT
      END:VCALENDAR
```

Attendee A accepts the changes from B

```
      BEGIN:VCALENDAR
      PRODID:-//ACME/DesktopCalendar//EN
      METHOD:REQUEST
      VERSION:2.0
      BEGIN:VEVENT
      ATTENDEE;ROLE=OWNER:Mailto:A@example.com
      ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
```

```
     ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:B@example.
com
     ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:C@example.
com
     DTSTAMP:19970613T190000Z
     DTSTART:19970701T160000Z
     DTEND:19970701T190000Z
     SUMMARY:Discuss the Merits of the election results - changed to
       suite B's schedule
     LOCATION:The Small Conference Room
     UID:www.acme.com-873970198738777@host.com
     SEQUENCE:1
     STATUS:CONFIRMED
     END:VEVENT
     END:VCALENDAR
```

A rejects B's counter proposal

```
     BEGIN:VCALENDAR
     PRODID:-//ACME/DesktopCalendar//EN
     METHOD:DECLINECOUNTER
     VERSION:2.0
     BEGIN:VEVENT
     ATTENDEE;ROLE=OWNER:Mailto:A@example.com
     ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
     ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:B@example.
com
     COMMENT:Sorry, I cannot change this meeting time
     UID:www.acme.com-873970198738777@host.com
     SEQUENCE:1
     DTSTAMP:19970614T190000Z
     END:VEVENT
```

4.2.5 Delegate An Event

When delegating an event request to another "Calendar User", the "delegator" MUST both update the "Organizer" with a "REPLY" and send a request to the "delegatee". There is currently no protocol limitation to delegation depth. It is possible for the original delegate to delegate the meeting to someone else, and so on. When a request is delegated from one "CUA" to another there are a number of responsibilities required of the "delegator". They MUST:

- Send an REPLY to the "Organizer" with their attendee/status property parameter set to "Delegated"

. Include the delegate as an additional attendee with the "Delegated-From" property parameter set to the delegator

. Include the original UID of the REQUEST

. The delegator MUST also send a copy of the original REQUEST to the delegate. The delegator modifies the request to include:

. The ATTENDEE/STATUS property parameter for the delegator (sender in this case) is set to "DELEGATED"

. ATTENDEE/DELEGATED-TO parameter is set to the address of the delegatee

. An ATTENDEE property is added for delegatee

As a rule, it is not required that the "delegatee" include the "delegator" in their "REPLY" method. However, it is strongly advised since this will inform the "delegator" whether their proxy plans to attend the meeting. If the "delegatee" declines the meeting, the "delegator" MAY elect to try and delegate the "REQUEST" to another "CUA". The process is the same.

Action	"Organizer"	Attendee
Initiate a meeting request	"A" sends a REQUEST message to "B" and	
Delegate: "C" delegates to "E"		"C" sends a REPLY to "A" with the ATTENDEE.STATUS parameter set to "DELEGATED" and with a new ATTENDEE property for "E". "E's" ATTENDEE DELEGATED-FROM property is set to "C". "C's" ATTENDEE.DELEGATED-TO property is set to "E". "C" sends REQUEST message to "E" with the original meeting request information. The

			ATTENDEE/STATUS property
			parameter for "C" is set
			to "DELEGATED" and the
			ATTENDEE/DELEGATED-TO
			parameter is set to
			the address of "E". An
			ATTENDEE property is
			added for "E" and the
			ATTENDEE/DELEGATED-FROM
			parameter is set to
			the address of "C".
Confirm meeting attendance			"E" sends REPLY message to "A" and optionally "C" with its ATTENDEE/STATUS property parameter set

			to "ACCEPTED"
Optional: Redistribute meeting to attendees	"A" sends REQUEST message to "B", "C" and "E". SEQUENCE number is now 1.		

Attendee "C" responds to meeting "Organizer" "A"

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:REPLY
VERSION:2.0
BEGIN:VEVENT
ATTENDEE;ROLE=ATTENDEE;STATUS=DELEGATED;DELEGATED-
 TO="Mailto:E@example.com":Mailto:C@example.com
UID:www.acme.com-873970198738777@host.com
SEQUENCE:0
REQUEST-STATUS:2.0;Success
DTSTAMP:19970611T190000Z
END:VEVENT
END:VCALENDAR
```

Attendee "C" delegates presence at the meeting to "E".

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:REQUEST
VERSION:2.0
BEGIN:VEVENT
ATTENDEE;ROLE=ORGANIZER:Mailto:A@example.com
ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
ATTENDEE;ROLE=DELEGATE;RSVP=TRUE;EXPECT=REQUEST;
 DELEGATED-FROM=_Mailto:C@example.com_:Mailto:E@example.com
ATTENDEE;ROLE=ATTENDEE;STATUS=DELEGATED;
 DELEGATED-TO="Mailto:E@example.com":Mailto:C@example.com
DTSTART:19970701T110000-0700
DTEND:19970701T113000-0700
SUMMARY:Phone Conference
UID:www.acme.com-873970198738777@host.com
SEQUENCE:0
STATUS:CONFIRMED
DTSTAMP:19970611T190000Z
END:VEVENT
END:VCALENDAR
```


4.2.6 Delegate Accepts the Meeting

To accept a delegated meeting, the delegate sends the following message to "A" and "C"

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:REPLY
VERSION:2.0
BEGIN:VEVENT
ATTENDEE;ROLE=ATTENDEE;STATUS=CONFIRMED;
 DELEGATED-FROM="Mailto:C@example.com":Mailto:E@example.com
UID:www.acme.com-873970198738777@host.com
SEQUENCE:1
REQUEST-STATUS:2.0;Success
```

```
DTSTAMP:19970614T190000Z
END:VEVENT
END:VCALENDAR
```

4.2.7 Delegate Declines the Meeting

In this example the delegate declines the meeting request and sets the "ATTENDEE" property "STATUS" parameter to DECLINED. The "Organizer" SHOULD resend the "REQUEST" to "C" with the status of the delegate set to DECLINED. This lets the "delegator" know that the "delegate" has declined and provides an opportunity to the "delegator" to either accept or delegate the request to another "Calendar User".

Response from "E" to "A" and "C".

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:REPLY
VERSION:2.0
BEGIN:VEVENT
ATTENDEE;ROLE=ATTENDEE;STATUS=DECLINED;
 DELEGATED-FROM="Mailto:C@example.com":Mailto:E@example.com
UID:www.acme.com-873970198738777@host.com
SEQUENCE:1
REQUEST-STATUS:2.0;Success
DTSTAMP:19970614T190000Z
END:VEVENT
END:VCALENDAR
```

"A" resends the "REQUEST" method to "C". "A" MAY also wish to express the fact that the item was delegated in the "COMMENT" property.

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
```



```
METHOD:REPLY
VERSION:2.0
BEGIN:VEVENT
ATTENDEE;ROLE=ATTENDEE;STATUS=DECLINED;
 DELEGATED-FROM="Mailto:C@example.com":Mailto:E@example.com
```

```
     ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:C@example.
com
     UID:www.acme.com-873970198738777@host.com
     SEQUENCE:2
     REQUEST-STATUS:2.0;Success
     DTSTAMP:19970614T200000Z
     COMMENT:DELEGATE (ATTENDEE Mailto:E@example.com) DECLINED YOUR
INVITATION
     END:VEVENT
     END:VCALENDAR
```

4.2.8 Forwarding an Event Request

The protocol does not prevent an "Attendee" from "forwarding" an "VEVENT" calendar component to other "Calendar Users". Forwarding differs from delegation in that the forwarded "Calendar User" (often referred to as a "Party Crasher") does not replace the forwarding "Calendar User". Implementations are not required to add the "Party Crasher" to the "Attendee" list and hence there is no guarantee that a "Party Crasher" will receive additional updates to the Event. The forwarding "Calendar User" SHOULD NOT add the "Party Crasher" to the attendee list.

4.2.9 Cancel A Group Event

Individual "A" requests a meeting between individuals "A", "B" and "C". Individual "B" declines attendance to the meeting. Individual "A" decides to cancel the meeting. The following table illustrates the sequence of messages that would be exchanged between these individuals.

Messages related to a previously canceled event ("SEQUENCE" property value is less than the "SEQUENCE" property value of the "CANCEL" message) or "VTODO" calendar component MUST be ignored.

Action	"Organizer"	Attendee
Initiate a meeting request	"A" sends a REQUEST message to "B" and "C"	
Decline the meeting request		"B" sends a REPLY message to "A" with its ATTENDEE STATUS para-set to "DECLINED".

Cancel the meeting	"A" sends a CANCEL message to "B" and "C"	

The example shows how "A" cancels the event.

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:CANCEL
VERSION:2.0
BEGIN:VEVENT
COMMENT:Mr. B cannot attend. I'll reschedule the meeting later.
UID:www.acme.com-873970198738777@host.com
SEQUENCE:0
DTSTAMP:19970613T190000Z
STATUS:CANCELLED
END:VEVENT
END:VCALENDAR
```

The "Organizer" of a meeting MAY "uninvite" or remove "Attendees" by sending a "CANCEL" method to only those "Attendees".

4.3 Busy Time Examples

Individual "A" requests busy time from individuals "B", "C" and "D". Individual "B" and "C" replies with busy time data to individual "A". Individual "D" does not support busy time requests and does not reply with any data.

The following table illustrates the sequence of messages that would be exchanged between these individuals.

Action	"Organizer"	Attendee
Initiate a busy time request	"A" sends a REQUEST message to "B" and and "C"	
Reply to the busy request with busy		"B" sends a REPLY message to "A" with

| time data | | busy time data |
+---+

4.3.1 Request Busy Time

"A" sends a BUSY-"REQUEST" to "B" and "C" for busy time


```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:REQUEST
VERSION:2.0
BEGIN:VFREEBUSY
ATTENDEE;ROLE=OWNER:Mailto:A@example.com
ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
ATTENDEE:Mailto:B@example.com
ATTENDEE:Mailto:C@example.com
DTSTAMP:19970613T190000Z
DTSTART:19970701T080000-0700
DTEND:19970701T200000-0700
UID:www.acme.com-873970198738777@host.com
END:VFREEBUSY
END:VCALENDAR
```

4.3.2 Reply To A Busy Time Request

"B" sends a "REPLY" method type of a "VFREEBUSY" calendar component to "A"

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:REPLY
VERSION:2.0
BEGIN:VFREEBUSY
ATTENDEE:Mailto:B@example.com
DTSTART:19970701T080000-0700
DTEND:19970701T200000-0700
UID:www.acme.com-873970198738777@host.com
FREEBUSY:19970701T090000-0700/PT1H,19970701T140000-0700/PT30H
DTSTAMP:19970613T190030Z
```

```
END:VFREEBUSY
END:VCALENDAR
```

B is busy from 09:00 to 10:00 and from 14:00 to 14:30.

4.4 Recurring Event and Time Zone Examples

4.4.1 A Recurring Event Spanning Time Zones

This event describes a weekly phone conference. The "Attendees" are each in a different time zone.

```
BEGIN:VCALENDAR

PRODID:-//ACME/DesktopCalendar//EN
METHOD:REQUEST
```



```
VERSION:2.0

BEGIN:VTIMEZONE
DAYLIGHT:TRUE
DTSTART:19970406T200000-0800
RRULE:FREQ=YEARLY;BYDAY=1SU;BYMONTH=4
TZNAME:PDT
TZOFFSET:-0700
END:VTIMEZONE

BEGIN:VTIMEZONE
DAYLIGHT:FALSE
DTSTART:19971026T200000-0700
RRULE:FREQ=YEARLY;BYDAY=-1SU;BYMONTH=10
TZNAME:PST
TZOFFSET:-0800
END:VTIMEZONE

BEGIN:VEVENT
ATTENDEE;ROLE=OWNER;STATUS=ACCEPTED:Mailto:A@example.com
ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:B@example.fr
ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:c@example.jp
DTSTAMP:19970613T190030Z
DTSTART:19970701T140000
```

```
DTEND:19970701T150000
RRULE:FREQ=WEEKLY;INTERVAL=20;WKST=SU;BYDAY=TU
RDATE:19970910T140000
EXDATE:19970909T140000
EXDATE:19971028T150000
SUMMARY:Weekly Phone Conference
UID:www.acme.com-873970198738777@host.com
SEQUENCE:0
STATUS:CONFIRMED
END:VEVENT

END:VCALENDAR
```

The "VTIMEZONE" components SHOULD appear in an iCalendar object containing recurring events. This is especially important if a recurring event has "Attendees" in different time zones. There MAY be multiple VTIMEZONE components in an iCalendar object, however, they MUST be used to define the same time zone. That is, there cannot be VTIMEZONE components describing both PST/PDT and EST/EDT at the component level in the same iCalendar object.

The first two components of this iCalendar object are the time zone components. The "DTSTART" date coincides with the first instance of the RRULE property.

The recurring meeting is defined in a particular time zone, presumably that of the originator. The client for each "Attendee" has the responsibility of determining the recurrence time in the "Attendee's" time zone.

The repeating event starts on Tuesday, July 1, 1997 at 2:00pm. Since no time zone is specified in the "DTSTART" property, the time zone component of PDT applies to the start and end times. "Attendee" B@example.fr is in France where the local time on this date is 7 hours later than PDT or 21:00. "Attendee" C@example.jp is in Japan where local time is 9 hours ahead of than UTC or Wednesday, July 2 at 07:00. The event repeats weekly on Tuesdays (in PST/PDT). The "RRULE" property results in 20 instances. The last instance falls on Tuesday, November 11, 1997 2:00pm PST. The "RDATE" property adds another instance: WED, 10-SEP-1997 21:00 GMT.

There are two exceptions to this recurring appointment. The first one is:

TUE, 09-SEP-1997 21:00 GMT
TUE, 09-SEP-1997 14:00 PDT
WED, 10-SEP-1997 07:00 JDT

and the second is:

TUE, 28-OCT-1997 22:00 GMT
TUE, 28-OCT-1997 14:00 PST
WED, 29-OCT-1997 07:00 JST

4.4.2 Modify A Recurring Instance

In this example the "Organizer" issues a recurring meeting. Later the "Organizer" changes an instance of the event by changing the "DTSTART" property. Note the use of "RECURRENCE-ID" property and "SEQUENCE" property in the second request.

Original Request:

```
BEGIN:VCALENDAR
METHOD:REQUEST
PRODID:-//RDU Software//NONSGML HandCal//EN
VERSION:2.0
BEGIN:VEVENT
CREATED:19970526T083000
UID:guid-1@host1.com
SEQUENCE:0
RRULE:FREQ=MONTHLY;BYMONTHDAY=1;UNTIL=19980901T210000Z
ATTENDEE;ROLE=OWNER:Mailto:A@example.com
ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:B@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:C@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:D@example.com
DESCRIPTION:IETF-C&S Conference Call
CLASS:PUBLIC
SUMMARY:IETF Calendaring Working Group Meeting
DTSTART:19970601T210000Z
DTEND:19970601T220000Z
LOCATION:Conference Call
```



```
DTSTAMP:19970526T083000
STATUS:CONFIRMED
```

```
END:VEVENT
END:VCALENDAR
```

The event request below is to change a time and create an exception. This creates an exception on July 3rd.

```
BEGIN:VCALENDAR
METHOD:REQUEST
PRODID:-//RDU Software//NONSGML HandCal//EN
VERSION:2.0
BEGIN:VEVENT
CREATED:19970526T083000
UID:guid-1@host1com
RECURRENCE-ID:19970701T210000Z
SEQUENCE:1
RRULE:FREQ=MONTHLY;BYMONTHDAY=1;UNTIL=19980901T210000Z
ATTENDEE;ROLE=OWNER:Mailto:A@example.com
ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:B@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:C@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:D@example.com
DESCRIPTION:IETF-C&S Conference Call
CLASS:PUBLIC
SUMMARY:IETF Calendaring Working Group Meeting
DTSTART:19970703T210000Z
DTEND:19970703T220000Z
LOCATION:Conference Call
DTSTAMP:19970626T093000
STATUS:CONFIRMED
END:VEVENT
END:VCALENDAR
```

4.4.3 Cancel A Recurring Instance

In this example the "Organizer" of a recurring event wishes to delete an instance. This is referred to as an "exception" to the recurring event.

```
BEGIN:VCALENDAR
METHOD:CANCEL
PRODID:-//RDU Software//NONSGML HandCal//EN
VERSION:2.0
BEGIN:VEVENT
UID:guid-1@host1.com
RECURRENCE-ID:19970801T210000Z
SEQUENCE:2
DTSTAMP:19970721T093000
STATUS:CANCELLED
```

```
     END:VEVENT
```



```
     END:VCALENDAR
```

4.4.4 Cancel An Exception

In the following example, the "Organizer" has created an exception (as in 4.4.3) and now wishes to cancel it. In this case a "CANCEL" method is sent with the specific "RECURRENCE-ID", "UID" and "SEQUENCE" properties of the exception. This same sequence MAY be used to decline a previously accepted modification to a recurring event (as in 4.4.2).

```
  BEGIN:VCALENDAR
  METHOD:CANCEL
  PRODID:-//RDU Software//NONSGML HandCal//EN
  VERSION:2.0
  BEGIN:VEVENT
  UID:guid-1@host1.com
  RECURRENCE-ID:19970801T210000Z
  SEQUENCE:2
  DTSTAMP:19970721T103000
  STATUS:CANCELLED
  END:VEVENT
  END:VCALENDAR
```

4.4.5 Cancel Recurring Event

In this example the "Organizer" wishes to cancel the entire recurring event and any child exceptions.

```
  BEGIN:VCALENDAR
  METHOD:CANCEL
  PRODID:-//RDU Software//NONSGML HandCal//EN
  VERSION:2.0
  BEGIN:VEVENT
  UID:guid-1@host1.com
  DTSTAMP:19970721T103000
  SEQUENCE:2
  STATUS:CANCELLED
```

```
END:VEVENT
END:VCALENDAR
```

4.4.6 Change All Future Instances

This example changes the meeting location from a conference call to Seattle starting Sept 1 and extends to all future instances.

```
BEGIN:VCALENDAR
METHOD:REQUEST
```



```
PRODID:-//RDU Software//NONSGML HandCal//EN
VERSION:2.0
BEGIN:VEVENT
CREATED:19970526T083000
UID:guid-1@host1.com
RECURRENCE-ID;THISANDFUTURE:19970901T210000Z
SEQUENCE:3
RRULE:FREQ=MONTHLY;BYMONTHDAY=1;UNTIL=19980901T210000Z
ATTENDEE;ROLE=OWNER:Mailto:A@example.com
ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:B@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:C@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:D@example.com
DESCRIPTION:IETF-C&S Conference Call
CLASS:PUBLIC
SUMMARY:IETF Calendaring Working Group Meeting
DTSTART:19970901T210000Z [SS2][SS3]
DTEND:19970901T220000Z
LOCATION:Building 32, Microsoft, Seattle, WA
DTSTAMP:19970526T083000
STATUS:CONFIRMED
END:VEVENT
END:VCALENDAR
```

4.4.7 Add A New Instance To A Recurring Event

This example adds a one-time additional instance to the recurring event. "Organizer" adds a second July meeting on the 15th.

```
BEGIN:VCALENDAR
METHOD:REQUEST
PRODID:-//RDU Software//NONSGML HandCal//EN
VERSION:2.0
BEGIN:VEVENT
CREATED:19970526T083000
UID:guid-1@host1.com
RECURRENCE-ID:19970715T210000Z
SEQUENCE:4
RRULE:FREQ=MONTHLY;BYMONTHDAY=1;UNTIL=19980901T210000Z
RDATE;VALUE=PERIOD:19970715T210000Z/19970715T220000Z
ATTENDEE;ROLE=OWNER:Mailto:A@example.com
ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:B@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:C@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:D@example.com
DESCRIPTION:IETF-C&S Conference Call
CLASS:PUBLIC
SUMMARY:IETF Calendaring Working Group Meeting
DTSTART:19970715T210000Z
DTEND:19970715T220000Z
LOCATION:Conference Call
DTSTAMP:19970629T093000
```



```
STATUS:CONFIRMED
END:VEVENT
END:VCALENDAR
```

4.4.8 Counter An Instance Of A Recurring Event

In this example one of the "Attendees" counters the "DTSTART" property of the proposed second July meeting.

```
BEGIN:VCALENDAR
METHOD:COUNTER
PRODID:-//RDU Software//NONSGML HandCal//EN
VERSION:2.0
BEGIN:VEVENT
CREATED:19970526T083000
UID:guid-1@host1.com
RECURRENCE-ID:19970715T210000Z
SEQUENCE:4
```

```
RRULE:FREQ=MONTHLY;BYMONTHDAY=1;UNTIL=19980901T210000Z
ATTENDEE;ROLE=OWNER:Mailto:A@example.com
ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:B@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:C@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:D@example.com
DESCRIPTION:IETF-C&S Conference Call
CLASS:PUBLIC
SUMMARY:IETF Calendaring Working Group Meeting
DTSTART:19970715T220000Z
DTEND:19970715T230000Z
LOCATION:Conference Call
COMMENT:May we bump this by an hour? I have a conflict
DTSTAMP:19970629T094000
END:VEVENT
END:VCALENDAR
```

4.4.9 Error Reply To A request

The following example illustrates a scenario where a meeting is proposed that contains a property that is not supported (in this case, the "RRULE" property).

Original Request:

```
BEGIN:VCALENDAR
METHOD:REQUEST
PRODID:-//RDU Software//NONSGML HandCal//EN
VERSION:2.0
BEGIN:VEVENT
CREATED:19970526T083000
UID:guid-1@host1.com
```



```
SEQUENCE:0
RRULE:FREQ=MONTHLY;BYMONTHDAY=1
ATTENDEE;ROLE=OWNER:Mailto:A@example.com
ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:B@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:C@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:D@example.com
DESCRIPTION:IETF-C&S Conference Call
CLASS:PUBLIC
```

```
SUMMARY:IETF Calendaring Working Group Meeting
DTSTART:19970601T210000Z
DTEND:19970601T220000Z
DTSTAMP:19970602T094000
LOCATION:Conference Call
STATUS:CONFIRMED
END:VEVENT
END:VCALENDAR
```

Response to indicate that RRULE is not supported:

```
BEGIN:VCALENDAR
PRODID:-//RDU Software//NONSGML HandCal//EN
METHOD:REPLY
VERSION:2.0
BEGIN:VEVENT
REQUEST-STATUS:2.8;Repeating event ignored. Scheduled as a single
  event;RRULE
UID:guid-1@host1.com
SEQUENCE:0
DTSTAMP:19970603T094000
END:VEVENT
END:VCALENDAR
```

4.5 Group To-do Examples

Individual "A" creates a group task in which individuals "A", "B", "C" and "D" will participate. Individual "B" confirms acceptance of the task. Individual "C" declines the task. Individual "D" tentatively accepts the task. The following table illustrates the sequence of messages that would be exchanged between these individuals. Individual "A" then issues a "REFRESH" method to obtain the status of the to-do from each participant. The response indicates the individual "Attendee's" completion status. The table below illustrates the message flow.

Action	"Organizer"	Attendee
Initiate a to-do request	"A" sends a REQUEST message to "B", "C", and "D"	

Accept the to-do request		"B" sends a REPLY message to "A" with its ATTENDEE STATUS para-set to "ACCEPTED".
Decline the to-do request		"C" sends a REPLY message to "A" with its ATTENDEE STATUS para-set to "DECLINED".
Tentatively accept the to-do request		"D" sends a REPLY message to "A" with its ATTENDEE STATUS para-set to "TENTATIVE".
Check attendee completion status	"A" sends a REFRESH message to "B" and "C" with current information.	
Attendee indicates percent complete		"B" sends a REPLY message indicating percent complete
Attendee indicates completion		"C" sends a REPLY message indicating completion

4.5.1 A VTODO Request

A sample "REQUEST" with for a "VTODO" calendar component that "A" sends to "B", "C", and "D".

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:REQUEST
VERSION:2.0
BEGIN:VTODO
ATTENDEE;ROLE=OWNER:Mailto:A@example.com
ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:B@example.
com
ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:C@example.
com
ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:D@example.
```

com
DTSTART:19970701T100000-0700
DUE:19970722T100000-0700
SUMMARY:Create the requirements document
UID:www.acme.com-873970198738777-00@host.com

SEQUENCE:0
DTSTAMP:19970717T200000Z
STATUS:CONFIRMED
END:VTODO
END:VCALENDAR

4.5.2 A VTODO Reply

Attendee "B" accepts the meeting.

BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:REPLY
VERSION:2.0
BEGIN:VTODO
ATTENDEE;STATUS=ACCEPTED:Mailto:B@example.com
UID:www.acme.com-873970198738777-00@host.com
COMMENT:I'll send you my input by e-mail
SEQUENCE:0
DTSTAMP:19970717T203000Z
REQUEST-STATUS:2.0;Success
END:VTODO
END:VCALENDAR

"B" could have declined the meeting or indicated tentative acceptance by setting the "ATTENDEE;STATUS=" property parameter sequence to DECLINED or TENTATIVE, respectively.

4.5.3 A VTODO Refresh

"A" requests status from all "Attendees".

BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN

```
     METHOD:REFRESH
     VERSION:2.0
     BEGIN:VTODO
     ATTENDEE;ROLE=OWNER:Mailto:A@example.com
     ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
     ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:B@example.
com
     ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:C@example.
com
     ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:D@example.
com
     UID:www.acme.com-873970198738777-00@host.com
     DTSTAMP:19970717T230000Z
     END:VTODO
     END:VCALENDAR
```


4.5.4 A Refresh Reply: Percent-Complete

A reply indicating that the task in being worked on and that "B" is 75% complete with "B's" part of the assignment.

```
     BEGIN:VCALENDAR
     PRODID:-//ACME/DesktopCalendar//EN
     METHOD:REPLY
     VERSION:2.0
     BEGIN:VTODO
     ATTENDEE;STATUS=IN-PROCESS:Mailto:B@example.com
     PERCENT-COMPLETE:75
     UID:www.acme.com-873970198738777-00@host.com
     DTSTAMP:19970717T233000Z
     SEQUENCE:0
     END:VTODO
     END:VCALENDAR
```

4.5.5 A Refresh Reply: Completed

A reply indicating that "C" finished with "C's" part of the assignment.

```
     BEGIN:VCALENDAR
```

```
PRODID:-//ACME/DesktopCalendar//EN
METHOD:REPLY
VERSION:2.0
BEGIN:VTODO
ATTENDEE;STATUS=COMPLETED:Mailto:C@example.com
UID:www.acme.com-873970198738777-00@host.com
DTSTAMP:19970717T233000Z
SEQUENCE:0
END:VTODO
END:VCALENDAR
```

4.5.6 An Updated VTODO Request

Owner "A" resends the "VTODO" calendar component. "A" set's the overall completion for the to-do at 40%.

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:REQUEST
VERSION:2.0
BEGIN:VTODO
ATTENDEE;ROLE=OWNER:Mailto:A@example.com
ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
ATTENDEE;STATUS=ACCEPTED;TYPE=INDIVIDUAL:Mailto:B@example.com
ATTENDEE;STATUS=COMPLETED;TYPE=INDIVIDUAL:Mailto:C@example.com
```



```
ATTENDEE;STATUS=TENTATIVE;TYPE=INDIVIDUAL:Mailto:D@example.com
DTSTART:19970701T100000-0700
DUE:19970722T100000-0700
SUMMARY:Create the requirements document
UID:www.acme.com-873970198738777-00@host.com
SEQUENCE:1
DTSTAMP:19970718T100000Z
STATUS:IN-PROGRESS
PERCENT-COMPLETE:40
END:VTODO
END:VCALENDAR
```

4.5.7 A Recurring VTODOs

The following examples relate to recurring "VTODO" calendar components.

4.5.7.1 Request for a Recurring VTODO

In this example "A" sends a recurring "VTODO" calendar component to "B" and "C".

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:REQUEST
VERSION:2.0
BEGIN:VTODO
ATTENDEE;ROLE=OWNER:Mailto:A@example.com
ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:B@example.
com
ATTENDEE;RSVP=TRUE;EXPECT=REQUEST;TYPE=INDIVIDUAL:Mailto:C@example.
com
RRULE:FREQ=MONTHLY;COUNT=10;BYDAY=1FR
DTSTART:19980101T100000-0700
DUE:19980103T100000-0700
SUMMARY:Send Status Reports to Area Managers
UID:www.acme.com-873970198738777-00@host.com
SEQUENCE:0
DTSTAMP:19970717T200000Z
STATUS:CONFIRMED
END:VTODO
END:VCALENDAR
```

4.5.7.2 Calculating due dates in recurring VTODOs

The due date in a recurring "VTODO" calendar component is either a fixed interval specified in the "REQUEST" method or specified specifically using the "RECURRENCE-ID" property. The former is

calculated by applying the difference between "DTSTART" and "DUE" properties and applying it to each of the start of each recurring instance. Hence, if the initial "VTODO" calendar component specifies a "DTSTART" property value of "19970701T190000Z" and a "DUE" property value of "19970801T190000Z" the interval of one day which could be

applied to each recurring instance of the "VTODO" calendar component.

4.5.7.3 Replying to an instance of a recurring VTODO

In this example "B" updates "A" on a single instance of the "VTODO" calendar component.

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:REPLY
VERSION:2.0
BEGIN:VTODO
ATTENDEE;STATUS=IN-PROCESS:Mailto:B@example.com
PERCENT-COMPLETE:75
UID:www.acme.com-873970198738777-00@host.com
DTSTAMP:19970717T233000Z
RECURRENCE-ID:19980101T100000-0700
SEQUENCE:0
END:VTODO
END:VCALENDAR
```

4.6 Journal Examples

The iCalendar object below describes a single journal entry for October 2, 1997. The "RELATED-TO" property references the phone conference event for which minutes were taken.

```
BEGIN:VCALENDAR
PROFILE:PUBLISH
PRODID:-//ACME/DesktopCalendar//EN
VERSION:2.0
BEGIN:VJOURNAL
DTSTART:19971002T200000Z
SUMMARY:Phone conference minutes
DESCRIPTION:The editors meeting was held on October 1, 1997.
  Details are in the attached document.
UID:0981234-1234234-2410@host.com
RELATED-TO:0981234-1234234-2402-35@host.com
ATTACH:ftp\://ftp.example.com/pub/ed/minutes100197.txt
END:VJOURNAL
END:VCALENDAR
```

4.7 Other Examples

4.7.1 Event Refresh

Refresh the event with "UID" property value of "guid-1-12345@host1.com":

```
BEGIN:VCALENDAR
PRODID:-//RDU Software//NONSGML HandCal//EN
METHOD:REFRESH
VERSION:2.0
BEGIN:VEVENT
ATTENDEE;ROLE=OWNER:Mailto:A@example.com
ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:B@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:C@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:D@example.com
UID: guid-1-12345@host1.com
DTSTAMP:19970603T094000
END:VEVENT
END:VCALENDAR
```

4.7.2 Bad RECURRENCE-ID

If an "Attendee" receives a request that references a "RECURRENCE-ID" property that can not be found, the "Attendee" SHOULD send a "REFRESH" method back to the "Organizer" for the latest copy of the event.

Action	"Organizer"	Attendee
Update an instance request	"A" sends a REQUEST message to "B"	
Attendee requests refresh because RecurrenceID was not found		"B" sends a REFRESH message to "A"
Refresh the entire Event	"A" sends the latest copy of the Event to "B"	

Request from "A":

```
BEGIN:VCALENDAR
METHOD:REQUEST
PRODID:-//RDU Software//NONSGML HandCal//EN
VERSION:2.0
BEGIN:VEVENT
UID:acme-12345@host1.com
SEQUENCE:3
RRULE:FREQ=WEEKLY
RDATE;VALUE=PERIOD:19970819T210000Z/199700819T220000Z
ATTENDEE;ROLE=OWNER:Mailto:A@example.com
ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:B@example.com
DESCRIPTION:IETF-C&S Conference Call
SUMMARY:IETF Calendaring Working Group Meeting
DTSTART:19970801T210000Z
DTEND:19970801T220000Z
DTSTAMP:19970726T083000
STATUS:CONFIRMED
END:VEVENT
END:VCALENDAR
```

"B" has the event with "UID" property"acme-12345@host1.com" but the "SEQUENCE" property value is "1" and the event does not have an instance at the specified recurrence time. This means that the "Owner" is either adding a new instance or that the new instance was added when "SEQUENCE" property value "2" of the event was generated. In either case, "B" needs a new copy of the event.

```
BEGIN:VCALENDAR
PRODID:-//RDU Software//NONSGML HandCal//EN
METHOD:REFRESH
VERSION:2.0
```

```
BEGIN:VEVENT
ATTENDEE;ROLE=OWNER:Mailto:A@example.com
ATTENDEE;ROLE=ATTENDEE;STATUS=ACCEPTED:Mailto:A@example.com
ATTENDEE;EXPECT=REQUEST:Mailto:B@example.com
UID:acme-12345@host1.com
DTSTAMP:19970603T094000
END:VEVENT
END:VCALENDAR
```

5 Application Protocol Fallbacks

5.1 Partial Implementation

Applications that support this memo are not required to support the entire protocol. The following describes how methods and properties SHOULD "fallback" in applications that do not support the complete

protocol. If a method or property is not addressed in this section, it MAY be ignored.

5.1.1 Event-Related Fallbacks

Method	Fallback
PUBLISH	Required.
CANCEL	Required.
REQUEST	PUBLISH
REPLY	Required.
DELEGATE	Reply with Not Supported.
REQUEST	Reply with Not Supported.
REPLY	Reply with Not Supported.
COUNTER	Reply with Not Supported
DECLINECOUNTER	Required if EVENT-COUNTER is implemented; otherwise reply with Not Supported.

iCalendar Property	Fallback
CALSCALE	Ignore; assume GREGORIAN.
GEO	Ignore.

PRODID	Ignore.
METHOD	Required as described in the Method list above.
SOURCE	Ignore
NAME	Ignore.
VERSION	Ignore.

Event-Related Components	Fallback
VFREEBUSY	Reply with Not Supported.
VALARM	Reply with Not Supported.
VTIMEZONE	Required if RRULE or RDATE is implemented; otherwise ignore and use local time.

Component Property	Fallback
ATTACH	Ignore.
ATTENDEE	Required if EVENT-REQUEST is not implemented; otherwise reply with Not Supported.
CATEGORIES	Required if in VALARM and VALARM is implemented, otherwise ignore.
CLASS	Ignore.
COMMENT	Ignore.
COMPLETED	Ignore.
CREATED	Ignore.
DAYLIGHT	Required if VTIMEZONE is implemented; otherwise Ignore.
DESCRIPTION	Required.
DELEGATED-FROM	Required if EVENT-DELEGATE is implemented; otherwise

	Ignore.
DELEGATED-TO	Required if EVENT-DELEGATE is implemented; otherwise Ignore.
DUE	Ignore.
DURATION	Reply with Not Supported.
DTSTAMP	Required.
DTSTART	Required.
DTEND	Required.
EXDATE	Ignore.
EXRULE	Ignore.
FREEBUSY	Reply with Not Supported.
LAST-MODIFIED	Ignore.

LOCATION	Required.
PRIORITY	Ignore.
RELATED-TO	Ignore.
RDATE	Ignore. If implemented, VTIMEZONE MUST also be implemented.
RRULE	Ignore. The first instance occurs on the DTStart property.
RECURRENCE-ID	Required if RRULE is implemented; otherwise Ignore.
REQUEST-STATUS	Required.
RESOURCES	Ignore.
SEQUENCE	Required.
STATUS	Ignore.
SUMMARY	Ignore.
TRANSP	Required if FREEBUSY is implemented; otherwise Ignore.
TZNAME	Required if VTIMEZONE is implemented; otherwise Ignore.
TZOFFSET	Required if VTIMEZONE is implemented; otherwise Ignore.
URL	Ignore.
UID	Required.
X-	Ignore.

5.1.2 To-Do-Related Fallbacks

Method	Fallback
PUBLISH	Required.
CANCEL	Required.
REQUEST	TODO-PUBLISH
REPLY	Required.

iCalendar Property	Fallback
CALSCALE	Ignore; assume GREGORIAN.
GEO	Ignore.
PRODID	Ignore.
METHOD	Required as described in the Method list above.
SOURCE	Ignore
NAME	Ignore.

VERSION	Ignore.

To-Do-Related Components	Fallback
VALARM	Reply with Not Supported.
VTIMEZONE	Required if RRULE or RDATE is implemented; otherwise ignore and use local time.

Component Property	Fallback
CALSCALE	Ignore; assume GREGORIAN.
GEO	Ignore.
PRODID	Ignore.
PROFILE	Required as described in the Method list above.
SOURCE	Ignore
NAME	Ignore.
VERSION	Assume "2.0".
Property	Fallback
ATTACH	Ignore.
ATTENDEE	Required if REQUEST is not implemented; otherwise ignore.
CATEGORIES	Ignore.
CLASS	Ignore.
COMMENT	Ignore.
COMPLETED	Required.
CREATED	Ignore.
DAYLIGHT	Required if VTIMEZONE is implemented; otherwise ignore.
DESCRIPTION	Required.
DUE	Required.
DURATION	Ignore. Reply with Not Supported.
DTSTAMP	Required.
DTSTART	Required.
EXDATE	Ignore. Reply with Not Supported.
EXRULE	Ignore. Reply with Not Supported.
LAST-MODIFIED	Ignore.
LOCATION	Ignore.
PERCENT-COMPLETE	Ignore
PRIORITY	Required.
RELATED-TO	Ignore.
RDATE	Ignore. If implemented, VTIMEZONE MUST also be implemented.
RRULE	Ignore. The first instance occurs on the DTStart property.
RESOURCES	Ignore.
SEQUENCE	Required.
STATUS	Required.
SUMMARY	Ignore.
TRANSP	Required if FREEBUSY is implemented; otherwise Ignore.
TZNAME	Required if VTIMEZONE is implemented; otherwise

	Ignore.
TZOFFSET	Required if VTIMEZONE is implemented; otherwise

	Ignore.
URL	Ignore.
UID	Required.
X-	Ignore.

5.1.3 Journal-Related Fallbacks

Method	Fallback
PUBLISH	Implementations MAY ignore the profile type. The REQUEST-STATUS "302;Request not supported" MUST be returned.
CANCEL	Implementations MAY ignore the profile type. The REQUEST-STATUS "302;Request not supported" MUST be returned.
REFRESH	Implementations MAY ignore the profile type. The REQUEST-STATUS "302;Request not supported" MUST be returned.

iCalendar Property	Fallback
CALSCALE	Ignore; assume GREGORIAN.
GEO	Ignore.
PRODID	Ignore.
METHOD	Required as described in the Method list above.
SOURCE	Ignore
NAME	Ignore.
VERSION	Ignore.

Journal-Related Components	Fallback
VTIMEZONE	Required if RRULE or RDATE is implemented; otherwise ignore and use local time.
CALSCALE	Ignore; assume GREGORIAN.
GEO	Ignore.
PRODID	Ignore.
METHOD	Required as described in the Method section above.

SOURCE	Ignore
NAME	Ignore.
VERSION	Assume "2.0".

Component Property	Fallback
ATTACH	Ignore.
ATTENDEE	Required if JOURNAL-REQUEST is implemented; otherwise ignore.
CATEGORIES	Ignore.
CLASS	Ignore.
COMMENT	Ignore.

CREATED	Ignore.
DAYLIGHT	Required if VTIMEZONE is implemented; otherwise Ignore.
DESCRIPTION	Required.
DTSTAMP	Required.
DTSTART	Required.
DTEND	Required.
EXDATE	Ignore.
EXRULE	Ignore.
LAST-MODIFIED	Ignore.
RELATED-TO	Ignore.
RDATE	Ignore. If implemented, VTIMEZONE MUST also be implemented.
RRULE	Ignore. The first instance occurs on the DTStart property.
SEQUENCE	Required.
STATUS	Ignore.
TRANSP	Required if FREEBUSY is implemented; otherwise ignore.
TZNAME	Required if VTIMEZONE is implemented; otherwise ignore.
TZOFFSET	Required if VTIMEZONE is implemented; otherwise ignore.
URL	Ignore.
UID	Required.
X-	Ignore.

5.2 Latency Issues

With a store-and-forward transport, it is possible for events to arrive out of sequence. That is, you MAY receive a "CANCEL" method prior to receiving the associated "REQUEST" for the calendar component. This section discusses a few of these scenarios.

5.2.1 Cancellation of an Unknown Calendar Component.

When a "CANCEL" method is received before the original "REQUEST" method the calendar will be unable to correlate the "UID" property of the cancellation with an existing calendar component. It is suggested that messages that can not be correlated that also contain non-zero sequence numbers be held and not discarded. Implementations MAY age them out if no other messages arrive with the same "UID" property value and a lower sequence number.

5.2.2 Unexpected Reply from an Unknown Delegate

When an "Attendee" delegates an item to another "Calendar User" they MUST send a "REPLY" method to the "Organizer" using the "ATTENDEE" properties to indicate the fact that the request was delegated and to whom the item was delegated. Hence it is possible for an "Organizer" to receive an "REPLY" from a "Calendar User" not listed as one of the

original "Attendees". The resolution is left to the implementation but it is expected that the calendaring software will either accept the reply or hold it until the related "REPLY" method is received from the "delegator". If the version of the "REPLY" method is out of date the "Organizer" SHOULD treat the message as a "STATUS-REQUEST" and update the delegate with the correct version.

5.3 Sequence Number

Under some conditions, a "CUA" MAY receive requests and replies with the same "SEQUENCE" property value. The "DTSTAMP" property is utilized as a tie-breaker when two items with the same "SEQUENCE" property value are evaluated. Furthermore, the "SEQUENCE" property is only incremented when one or more of the following properties changes:

. DTSTART

. DTEND

. RDATE

. RRULE

. EXDATE

. EXRULE

. DUE (for VTODO components)

. and possibly LOCATION

6 Security Considerations

This memo outlines an abstract transport protocol which will be bound to a real-time transport, a store-and-forward transport, and perhaps other transports. The transport protocol will be responsible for providing facilities for authentication and encryption using standard Internet mechanisms that are mutually understood between the sender and receiver.

6.1 Security Threats

6.1.1 Spoofing the "Organizer"

In this memo, the "Organizer" is the only person authorized to make changes to an existing "VEVENT", "VTODO", "VJOURNAL" calendar component and redistribute the updates to the "Attendees". An iCalendar object that maliciously changes or cancels an existing "VEVENT", "VTODO" or "VJOURNAL" calendar component MAY be constructed by someone other than the "Organizer" and sent to the "Attendees".

6.1.2 Spoofing the "Attendee"

In this memo, an "Attendee" of a "VEVENT", "VTODO", "VJOURNAL" calendar component is the only person authorized to update any parameter associated with their "ATTENDEE" property and send it to the "Organizer". An iCalendar object that maliciously changes the "ATTENDEE" parameters MAY be constructed by someone other than the real "Attendee" and sent to the "Organizer".

6.1.3 Eavesdropping

The iCalendar object is constructed with human-readable clear text. Any information contained in an iCalendar object MAY be read and/or changed by unauthorized persons while the object is in transit.

6.1.4 Flooding a Calendar

Implementations MAY provide a means to automatically incorporate "REQUEST" methods into a calendar. This presents the opportunity for a calendar to be flooded with requests, which effectively block all the calendar's free time.

6.1.5 Procedural Alarms

The "REQUEST" methods for "VEVENT" and "VTODO" calendar components MAY contain "VALARM" calendar components. The "VALARM" calendar component MAY be of type PROCEDURE and MAY have an attachment containing some sort of executable program. Implementations that incorporate these types of alarms are subject to any virus or malicious attack that MAY occur as a result of executing the attachment.

6.2 Recommendations

For an application where the information is sensitive or critical and the network is subject is subject to a high probability of attack, iTIP transactions SHOULD be secured. This MAY be accomplished using public key technology, specifically Security Multiparts for MIME [RFC1847] in the iTIP transport binding. This helps mitigate the threats of spoofing, eavesdropping and malicious changes in transit.

6.2.1 Use of [RFC1847] to secure iTIP transactions

iTIP transport bindings SHOULD provide a mechanism based on Security Multiparts for MIME [RFC1847] to enable authentication of the sender's identity, and privacy and integrity of the data being transmitted. This allows the receiver of a signed iCalendar object to verify the identity of the sender. This sender MAY then be correlated

to an "ATTENDEE" property in the iCalendar object. If the correlation is made and the sender is authorized to make the requested change or update then the operation MAY proceed. It also allows the message to be encrypted to prevent unauthorized reading of the message contents in transit. iTIP transport binding documents describe this process in detail.

6.2.2 Implementation Controls

The threat of flooding a calendar SHOULD be mitigated by a calendar system that uses this protocol by providing controls that MAY be used to limit the acceptable sources for iTIP transactions, and perhaps the size of messages and volume of traffic, by source.

The threat of malicious procedural alarms SHOULD be mitigated by a calendar system that uses this protocol by providing controls that MAY be used to disallow procedural alarms in iTIP transactions and/or remove all alarms from the object before delivery to the recipient.

7 Acknowledgments

A hearty thanks to the following individuals who have participated in the drafting, review and discussion of this memo:

Anik Ganguly, Bruce Kahn, John Noerenberg, Leo Parker, John Rose, Vinod Seraphin, Richard Shusterman, Derik Stenerson, John Sun, Kevin Tsurutome.

8 Bibliography

[ICAL] "Internet Calendaring and Scheduling Core Object Specification - iCalendar", Internet-Draft, October 1997, ftp://ftp.ietf.org/internet-drafts/draft-ietf-calsch-ical-03.txt.

[ICMS] "Internet Calendaring Model Specification", Internet-Draft, October 1997, ftp://ftp.ietf.org/internet-drafts/draft-ietf-calsch-mod-01.txt.

[ID-UTF8] "UTF-8, a transformation format of Unicode and ISO 10646", Internet-Draft, July 1996, ftp://ftp.ietf.org/internet-drafts/draft-yergeau-utf8-01.txt.

[IMIP] "iCalendar Message-Based Interoperability Protocol - iMIP", Internet-Draft, October 1997, ftp://ftp.ietf.org/internet-drafts/draft-ietf-calsch-imip-01.txt.

[ISO8601] "Data elements and interchange formats - information interchange - Representation of dates and times", ISO 8601, 1996-06-15, +1 (212) 642-4900 for ANSI Sales.

[VCAL] "vCalendar - The Electronic Calendaring and Scheduling Format - Version 1.0", Versit Consortium, September 18, 1996, http://www.imc.org/pdi/vcal-10.doc.

[VCARD] "vCard - The Electronic Business Card Exchange Format - Version 2.1", Versit Consortium, September 18, 1996, http://www.imc.org/pdi/vcard-21.doc.

[RFC821] Postel, "Simple Mail Transfer Protocol", RFC 821, organization name, November 1996, http://ds.internic.net/rfc/rfc821.txt.

[RFC1983] "Internet Users' Glossary", RFC 1983, August 1996, http://ds.internic.net/rfc/rfc1983.txt.

[RFC2119] "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997, http://ds.internic.net/rfc/rfc2119.txt.

[RFC2045] N. Freed and N. Borenstein, "Multipurpose Internet Mail Extensions - Part One - Format of Internet Message Bodies", RFC 2045, Innosoft, First Virtual, November 1996, http://ds.internic.net/rfc/rfc2045.txt.

[RFC2046] N. Freed and N. Borenstein, "Multipurpose Internet Mail Extensions - Part Two - Media Types", RFC 2046, Innosoft, First Virtual, November 1996, http://ds.internic.net/rfc/rfc2046.txt.

[UNICODE] The Unicode Consortium, "The Unicode Standard -Version 2.0", Addison-Wesley Developers Press, July 1996. UTF-8 is described in section A-2.

[US-ASCII] Coded Character Set--7-bit AmeriMAYStandard Code for Information Interchange, ANSI X3.4-1986.

9 Authors Addresses

The following address information is provided in a MIME-VCARD, Electronic Business Card, format.

The authors of this draft are:

```
BEGIN:VCARD
FN:Frank Dawson
ORG:Lotus Development Corporation
```

```
ADR;WORK;POSTAL;PARCEL:;;6544 Battleford Drive;Raleigh;NC;27613-
  3502;USA
TEL;WORK;MSG:+1-919-676-9515
TEL;WORK;FAX:+1-919-676-9564
EMAIL;INTERNET:Frank_Dawson@Lotus.com
URL:http://home.earthlink.net/~fdawson
END:VCARD

BEGIN:VCARD
FN:Ross Hopson
ORG:On Technology, Inc.
ADR;WORK;POSTAL;PARCEL:Suite 1600;;434 Fayetteville St.
  Mall, Two Hannover Square;Raleigh;NC;27601
TEL;WORK;MSG:+1-919-890-4036
```



```
TEL;WORK;FAX:+1-919-890-4100
EMAIL;INTERNET:rhopson@on.com
END:VCARD

BEGIN:VCARD
FN:Steve Mansour
ORG:Netscape Communications Corporation
ADR;WORK;POSTAL;PARCEL:;;501 East Middlefield Road;Mountain
  View;CA;94043;USA
TEL;WORK;MSG:+1-415-937-2378
TEL;WORK;FAX:+1-415-428-4059
EMAIL;INTERNET:sman@netscape.com
END:VCARD

BEGIN:VCARD
FN:Steve Silverberg
ORG:Microsoft Corporation
ADR;WORK;POSTAL;PARCEL:;;One Microsoft Way;
Redmond;WA;98052-6399;USA
TEL;WORK;MSG:+1-425-936-9277
TEL;WORK;FAX:+1-425-936-8019
EMAIL;INTERNET:stevesil@Microsoft.com
END:VCARD
```

The iCalendar object is a result of the work of the Internet Engineering Task Force Calendaring and scheduling Working Group. The chairman of that working group is:

```
BEGIN:VCARD
FN:Anik Ganguly
ORG:Campbel Services, Inc.
ADR;WORK;POSTAL;PARCEL:10 Floor;;21700 Northwestern
  Highway;Southfield;MI;48075;USA
TEL;WORK;MSG:+1-248-559-5955
TEL;WORK;FAX:+1-248-559-5034
EMAIL;INTERNET:anik@ontime.com
END:VCARD
```

10 Full Copyright Statement

"Copyright (C) The Internet Society (date). All Rights Reserved.

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 implmentation 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 paragraph are included on all such copies and derivative works. However, this document itself MAY NOT be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for

copyrights defined in the Internet Standards process 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 the Internet Society or its successors or assigns.

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