Network Working Group   S. Silverberg
Request for Comments: 2446   Microsoft
Category: Standards Track   S. Mansour
                                                             Netscape
                                                             F. Dawson
                                                                Lotus
                                                             R. Hopson
                                                        ON Technologies
                                                        November 1998

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

Status of this Memo

Copyright Notice

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.

   This document specifies an Internet protocol based on the iCalendar
   object specification that provides scheduling interoperability
   between different calendar systems. The Internet protocol is called
   the "iCalendar Transport-Independent Interoperability Protocol
   (iTIP)".

iTIP complements the iCalendar object specification by adding
semantics for group scheduling methods commonly available in current
calendar systems. These scheduling methods permit two or more
calendar systems to perform transactions such as publish, schedule,
reschedule, respond to scheduling requests, negotiation of changes or
cancel iCalendar-based calendar components.

iTIP is defined independent of the particular transport used to
transmit the scheduling information. Companion memos to iTIP provide
bindings of the interoperability protocol to a number of Internet
protocols.

Table of Contents

1 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 transport bindings between systems that use this protocol.

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

1.1 Formatting Conventions

   In order to refer to elements of the calendaring and scheduling
   model, core object or interoperability protocol defined in [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 interpreted as described in [RFC-2119].

   Calendaring and scheduling roles 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" (CU)
   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. The related documents are:

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

    [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 ITIP Roles and Transactions

ITIP defines methods for exchanging [iCAL] objects for the purposes of group calendaring and scheduling between "Calendar Users" (CUs). CUs take on one of two roles in iTIP. The CU who initiates an exchange takes on the role of "Organizer". For example, the CU who proposes a group meeting is the "Organizer". The CUs asked to participate in the group meeting by the "Organizer" take on the role of "Attendee". Note that "role" is also a descriptive parameter to the _ATTENDEE_ property. Its use is to convey descriptive context to an "Attendee" such as "chair", "req-participant" or "non-participant" and has nothing to do with the calendaring workflow.

The ITIP methods are listed below and their usage and semantics are
defined 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 require the receiver to respond using           |
|                | 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.                                   |
|                |                                                      |
| ADD            | Add one or more instances to an existing             |
|                | VEVENT, VTODO, or VJOURNAL.                           |
|                |                                                      |
| CANCEL         | Cancel one or more instances of an existing          |
|                | VEVENT, VTODO, or VJOURNAL.                           |
|                |                                                      |
| 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-       | Used by the "Organizer" to decline the proposed      |
| COUNTER        | counter-proprosal.                                   |
+================+======================================================+
```

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

```
+================+===================================================+
| Originator     | Methods                                           |
|================+===================================================|
| Organizer      | PUBLISH, REQUEST, ADD, CANCEL, DECLINECOUNTER     |
|                |                                                   |
| Attendee       | REPLY, REFRESH, COUNTER                            |
|                | REQUEST only when delegating                      |
+================+===================================================+
```

Note that for some calendar component types, the allowable methods
are a subset of the above set.

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
above. The Transport Protocol defines how the iCalendar objects are
sent between the sender and receiver. This document focuses on the
Application Protocol. Binding documents such as [iMIP] focus on the
Transport Protocol.

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

```
+----------+                      +----------+
|          |         iTIP         |          |
|  Sender  |<-------------------->| Receiver |
|          |                      |          |
+----------+                      +----------+
```
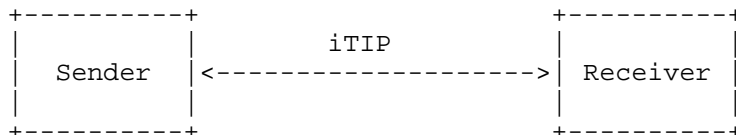
There are several variations of this diagram in which the Sender and
Receiver take on various roles of a "Calendar User Agent" (CUA) or a
"Calendar Service" (CS).

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.

```
+-------------------------------------------+
|                   iTIP                    |
+-------------------------------------------+
|Real-time | Store-and-Fwd | Other         |
|Transport | Transport     | Transports... |
+-------------------------------------------+
```

2.1 Application Protocol

   In the iTIP model, a calendar entry is created and managed by an
   "Organizer". The "Organizer" interacts with other CUs by sending one
   or more of the iTIP messages listed above. "Attendees" use the
   "REPLY" method to communicate their status. "Attendees" do not make
   direct changes to the master calendar entry. They can, however, use
   the "COUNTER" method to suggest changes to the "Organizer". In any
   case, the "Organizer" has complete control over the master calendar
   entry.

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" sets the "STATUS" property to the
   appropriate value for each calendar entry.

   The state of a particular "Attendee" relative to an entry is defined
   by the "partstat" parameter in the "ATTENDEE" property for each
   "Attendee". When an "Organizer" issues the initial entry, "Attendee"
   status is unknown. The "Organizer" specifies this by setting the
   "partstat" parameter to "NEEDS-ACTION". Each "Attendee" modifies
   their "ATTENDEE" property "partstat" parameter to an appropriate
   value as part of a "REPLY" message sent back to the "Organizer".

2.1.2 Delegation

   Delegation is defined as the process by which an "Attendee" grants
   another CU (or several CUs) the right to attend on their behalf. The
   "Organizer" is made aware of this change because the delegating
   "Attendee" informs the "Organizer". These steps are detailed in the
   REQUEST method section.

2.1.3 Acting on Behalf of other Calendar Users

   In many organizations one user will act on behalf of another to
   organize and/or respond to meeting requests. ITIP provides two
   mechanisms that support these activities.

   First, the "Organizer" is treated as a special entity, separate from
   "Attendees". All responses from "Attendees" flow to the "Organizer",
   making it easy to separate a calendar user organizing a meeting from
   calendar users attending the meeting. Additionally, iCalendar
   provides descriptive roles for each "Attendee". For instance, a role
   of "chair" may be ascribed to one or more "Attendees". The "chair"
   and the "Organizer" may or may not be the same calendar user. This
   maps well to scenarios where an assistant may manage meeting
   logistics for another individual who chairs a meeting.

   Second, a "sent-by" parameter may be specified in either the
   "Organizer" or "Attendee" properties. When specified, the "sent-by"
   parameter indicates that the responding CU acted on behalf of the
   specified "Attendee" or "Organizer".

2.1.4 Component Revisions

   The "SEQUENCE" property is used by the "Organizer" to indicate
   revisions to the calendar component. The rules for incrementing the
   "SEQUENCE" number are defined in [iCAL]. For clarity, these rules are
   paraphrased here in terms of how they are applied in [iTIP]. For a
   given "UID" in a calendar component:

   . For the "PUBLISH" and "REQUEST" methods, the "SEQUENCE" property
      value is incremented according to the rules defined in [iCAL].

   . The "SEQUENCE" property value MUST be incremented each time the
      "Organizer" uses the "ADD" or "CANCEL" methods.

   . The "SEQUENCE" property value MUST NOT be incremented when using
      "REPLY", "REFRESH", "COUNTER", "DECLINECOUNTER", or when sending a
      delegation "REQUEST".

   In some circumstances the "Organizer" may not have received responses
   to the final revision sent out. In this situation, the "Organizer"
   may wish to send an update "REQUEST", and set "RSVP=TRUE" for all
   "Attendees", so that current responses can be collected.

The value of the "SEQUENCE" property contained in a response from an
"Attendee" may not always match the "Organizer's" revision.
Implementations may choose to have the CUA indicate to the CU that
the response is to an entry that has been revised and allow the CU to
decide whether or not to accept the response.

2.1.5 Message Sequencing

CUAs that handle the [iTIP] application protocol must often correlate
a component in a calendar store with a component received in the
[iTIP] message. For example, an event may be updated with a later
revision of the same event. To accomplish this, a CUA must correlate
the version of the event already in its calendar store with the
version sent in the [iTIP] message. In addition to this correlation,
there are several factors that can cause [iTIP] messages to arrive in
an unexpected order. That is, an "Organizer" could receive a reply
to an earlier revision of a component AFTER receiving a reply to a
later revision.

To maximize interoperability and to handle messages that arrive in an
unexpected order, use the following rules:

1. The primary key for referencing a particular iCalendar component
    is the "UID" property value. To reference an instance of a
    recurring component, the primary key is composed of the "UID" and
    the "RECURRENCE-ID" properties.

2. The secondary key for referencing a component is the "SEQUENCE"
    property value. For components where the "UID" is the same, the
    component with the highest numeric value for the "SEQUENCE"
    property obsoletes all other revisions of the component with
    lower values.

3. "Attendees" send "REPLY" messages to the "Organizer". For
    replies where the "UID" property value is the same, the value of
    the "SEQUENCE" property indicates the revision of the component
    to which the "Attendee" is replying. The reply with the highest
    numeric value for the "SEQUENCE" property obsoletes all other
    replies with lower values.

4. In situations where the "UID" and "SEQUENCE" properties match,
    the "DTSTAMP" property is used as the tie-breaker. The component
    with the latest "DTSTAMP" overrides all others. Similarly, for
    "Attendee" responses where the "UID" property values match and
    the "SEQUENCE" property values match, the response with the
    latest "DTSTAMP" overrides all others.

Hence, CUAs must persist the following component properties: "UID",
"RECURRENCE-ID", "SEQUENCE", and "DTSTAMP". Furthermore, for each
"ATTENDEE" property of a component CUAs must persist the "SEQUENCE"
and "DTSTAMP" property values associated with the "Attendee's"
response.

3 Application Protocol Elements

ITIP 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	No	No
Cancel	Yes	Yes	Yes	No
Add	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 components and
properties. Some components and properties are required, some are
optional and others are excluded. The restrictions are expressed in
this document using a simple "restriction table". The first column
indicates the name of a component or property. Properties of the
iCalendar object are not indented. Properties of a component are
indented. The second column contains "MUST" if the component or
property must be present, "MAY" if the component or property is
optional, and "NOT" if the component or property must not be present.
Entries in the second column sometimes contain comments for further
clarification.

3.1 Common Component Restriction Tables

   The restriction table below applies to properties of the iCalendar
   object. That is, the properties at the outermost scope. The presence
   column uses the following values to assert whether a property is
   required, is optional and the number of times it may appear in the
   iCalendar object.

   Presence Value   Description
   ----
   1   One instance MUST be present
   1+   At least one instance MUST be present
   0   Instances of this property Must NOT be present
   0+   Multiple instances MAY be present
   0 or 1   Up to 1 instance of this property MAY be present

   The tables also call out "X-PROPERTY" and "X-COMPONENT" to show
   where vendor-specific properties and components can appear. The
   tables do not lay out the restrictions of property parameters. Those
   restrictions are defined in [iCAL].

   Component/Property Presence
   ------------------- --
   CALSCALE   0 or 1
   PRODID   1
   VERSION   1   Value MUST be "2.0"
   X-PROPERTY   0+

   DateTime values MAY refer to a "VTIMEZONE" component. The property
   restrictions in the table below apply to any "VTIMEZONE" component in
   an ITIP message.

   Component/Property Presence
   ------------------- --
   VTIMEZONE   0+   MUST be present if any date/time refers
                                to timezone
      DAYLIGHT   0+   MUST be one or more of either STANDARD or
                                DAYLIGHT
         COMMENT   0 or 1
         DTSTART   1   MUST be local time format
         RDATE   0+   if present RRULE MUST NOT be present
         RRULE   0+   if present RDATE MUST NOT be present
         TZNAME   0 or 1
         TZOFFSET   1
         TZOFFSETFROM 1
         TZOFFSETTO   1

```
       X-PROPERTY      0+
   LAST-MODIFIED       0 or 1
   STANDARD            0+       MUST be one or more of either STANDARD or
                               DAYLIGHT
       COMMENT         0 or 1
       DTSTART         1        MUST be local time format
       RDATE           0+       if present RRULE MUST NOT be present
       RRULE           0+       if present RDATE MUST NOT be present
       TZNAME          0 or 1
       TZOFFSETFROM    1
       TZOFFSETTO      1
       X-PROPERTY      0+
   TZID                1
   TZURL               0 or 1
   X-PROPERTY          0+
```

   The property restrictions in the table below apply to any "VALARM"
   component in an ITIP message.

```
   Component/Property  Presence
   ------------------  ----------------------------------------------
   VALARM              0+
       ACTION          1
       ATTACH          0+
       DESCRIPTION     0 or 1
       DURATION        0 or 1  if present REPEAT MUST be present
       REPEAT          0 or 1  if present DURATION MUST be present
       SUMMARY         0 or 1
       TRIGGER         1
       X-PROPERTY      0+
```

3.2 Methods for VEVENT Calendar Components

   This section defines the property set restrictions for the method
   types that are applicable to the "VEVENT" calendar component. Each
   method is defined using a table 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 may degrade the event to view it as an        |
|                | PUBLISH.                                               |
|                |                                                       |
| REPLY          | Reply to an event request. Clients may set their      |
|                | status ("partstat") to ACCEPTED, DECLINED,            |
|                | TENTATIVE, or DELEGATED.                               |
|                |                                                       |
| ADD            | Add one or more instances to an existing event.       |
|                |                                                       |
| CANCEL         | Cancel one or more instances of an existing           |
|                | event.                                                |
|                |                                                       |
| REFRESH        | A request is sent to an "Organizer" by an             |
|                | "Attendee" asking for the latest version of an        |
|                | event to be resent to the requester.                  |
|                |                                                       |
| COUNTER        | Counter a REQUEST with an alternative proposal,       |
|                | Sent by an "Attendee" to the "Organizer".             |
|                |                                                       |
| DECLINECOUNTER | Decline a counter proposal. Sent to an                |
|                | "Attendee" by the "Organizer".                        |
+================+=======================================================+
```

3.2.1 PUBLISH

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

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

```
Component/Property  Presence
------------------  ---------------------------------------------
METHOD              1         MUST equal "PUBLISH"
VEVENT              1+
    DTSTAMP         1
    DTSTART         1
    ORGANIZER       1
    SUMMARY         1         Can be null.
    UID             1
    RECURRENCE-ID   0 or 1    only if referring to an instance of a
                              recurring calendar component.  Otherwise
                              it MUST NOT be present.
    SEQUENCE        0 or 1    MUST be present if value is greater than
                              0, MAY be present if 0
    ATTACH          0+
    CATEGORIES      0 or 1    This property may contain a list of
                              values
    CLASS           0 or 1
    COMMENT         0 or 1
    CONTACT         0+
    CREATED         0 or 1
    DESCRIPTION     0 or 1    Can be null
    DTEND           0 or 1    if present DURATION MUST NOT be present
    DURATION        0 or 1    if present DTEND MUST NOT be present
    EXDATE          0+
    EXRULE          0+
    GEO             0 or 1
    LAST-MODIFIED   0 or 1
    LOCATION        0 or 1
    PRIORITY        0 or 1
    RDATE           0+
    RELATED-TO      0+
    RESOURCES       0 or 1 This property MAY contain a list of values
    RRULE           0+
    STATUS          0 or 1 MAY be one of TENTATIVE/CONFIRMED/CANCELLED
    TRANSP          0 or 1
    URL             0 or 1
    X-PROPERTY      0+

    ATTENDEE        0
    REQUEST-STATUS  0

VALARM              0+
VFREEBUSY           0
VJOURNAL            0
```

```
VTODO                   0
VTIMEZONE               0+      MUST be present if any date/time refers to
                                a timezone
X-COMPONENT             0+
```

3.2.2 REQUEST

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

   . Invite "Attendees" to an event;
   . Reschedule an existing event;
   . Response to a REFRESH request;
   . 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;
   . Forward a "VEVENT" to another uninvited CU.
   . For an existing "VEVENT" calendar component, delegate the role of
      "Attendee" to another CU;
   . For an existing "VEVENT" calendar component, changing the role of
      "Organizer" to another CU.

   The "Organizer" originates the "REQUEST". The recipients of the
   "REQUEST" method are the CUs invited to the event, the "Attendees".
   "Attendees" use the "REPLY" method to convey attendance status to the
   "Organizer".

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

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

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

```
Component/Property  Presence
-------------------------------------------------------------------
METHOD               1        MUST be "REQUEST"
VEVENT               1+       All components MUST have the same UID
    ATTENDEE         1+
    DTSTAMP          1
    DTSTART          1
    ORGANIZER        1
    SEQUENCE         0 or 1   MUST be present if value is greater than 0,
                              MAY be present if 0
    SUMMARY          1        Can be null
    UID              1

    ATTACH           0+
    CATEGORIES       0 or 1   This property may contain a list of values
    CLASS            0 or 1
    COMMENT          0 or 1
    CONTACT          0+
    CREATED          0 or 1
    DESCRIPTION      0 or 1   Can be null
    DTEND            0 or 1   if present DURATION MUST NOT be present
    DURATION         0 or 1   if present DTEND MUST NOT be present
    EXDATE           0+
    EXRULE           0+
    GEO              0 or 1
    LAST-MODIFIED    0 or 1
    LOCATION         0 or 1
    PRIORITY         0 or 1
    RDATE            0+
    RECURRENCE-ID    0 or 1   only if referring to an instance of a
                              recurring calendar component.  Otherwise it
                              MUST NOT be present.
    RELATED-TO       0+
    REQUEST-STATUS   0+
    RESOURCES        0 or 1   This property MAY contain a list of values
    RRULE            0+
    STATUS           0 or 1   MAY be one of TENTATIVE/CONFIRMED
    TRANSP           0 or 1
    URL              0 or 1
    X-PROPERTY       0+

VALARM               0+
VTIMEZONE            0+       MUST be present if any date/time refers to
                              a timezone
X-COMPONENT         0+
```

```
VFREEBUSY            0
VJOURNAL             0
VTODO                0
```

3.2.2.1 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
   its 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" (or "DTSTAMP") 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.

3.2.2.2 Updating 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.

   The "Organizer" of an event may also send unsolicited "REQUEST"
   methods. The unsolicited "REQUEST" methods may be used to update the
   details of the event without rescheduling it, to update the
   "partstat" parameter of "Attendees", or to reconfirm the event.

3.2.2.3 Delegating an Event to another CU

   Some calendar and scheduling systems allow "Attendees" to delegate
   their presence at an event to another calendar user. ITIP supports
   this concept using the following workflow. Any "Attendee" may
   delegate their right to participate in a calendar VEVENT to another
   CU. The implication is that the delegate participates in lieu of the
   original "Attendee"; NOT in addition to the "Attendee". The delegator
   MUST notify the "Organizer" of this action using the steps outlined
   below. Implementations may support or restrict delegation as they
   see fit. For instance, some implementations may restrict a delegate
   from delegating a "REQUEST" to another CU.

The "Delegator" of an event forwards the existing "REQUEST" to the
"Delegate". The "REQUEST" method MUST include an "ATTENDEE" property
with the calendar address of the "Delegate". The "Delegator" MUST
also send a "REPLY" method to the "Organizer" with the "Delegator's"
"ATTENDEE" property "partstat" parameter value set to "delegated". In
addition, the "delegated-to" parameter MUST be included with the
calendar address of the "Delegate".

In response to the request, the "Delegate" MUST send a "REPLY" method
to the "Organizer" and optionally, to the "Delegator". The "REPLY"
method " SHOULD include the "ATTENDEE" property with the "delegated-
from" parameter value of the "Delegator's" calendar address.

The "Delegator" may continue to receive updates to the event even
though they will not be attending. This is accomplished by the
"Delegator" setting their "role" attribute to " NON-PARTICIPANT" in
the "REPLY" to the "Organizer"

3.2.2.4 Changing the Organizer

The situation may arise where the "Organizer" of a VEVENT is no
longer able to perform the "Organizer" role and abdicates without
passing on the "Organizer" role to someone else. When this occurs the
"Attendees" of the VEVENT may use out-of-band mechanisms to
communicate the situation and agree upon a new "Organizer". The new
"Organizer" should then send out a new "REQUEST" with a modified
version of the VEVENT in which the "SEQUENCE" number has been
incremented and value of the "ORGANIZER" property has been changed to
the calendar address of the new "Organizer".

3.2.2.5 Sending on Behalf of the Organizer

There are a number of scenarios that support the need for a calendar
user to act on behalf of the "Organizer" without explicit role
changing. This might be the case if the CU designated as "Organizer"
was sick or unable to perform duties associated with that function.
In these cases iTIP supports the notion of one CU acting on behalf of
another. Using the "sent-by" parameter, a calendar user could send an
updated "VEVENT" REQUEST. In the case where one CU sends on behalf of
another CU, the "Attendee" responses are still directed back towards
the CU designated as "Organizer".

3.2.2.6 Forwarding to An Uninvited CU

An "Attendee" invited to an event may invite another uninvited CU to
the event. The invited "Attendee" accomplishes this by forwarding the
original "REQUEST" method to the uninvited CU. The "Organizer"
decides whether or not the uninvited CU is added to the attendee

list. If the "Organizer" decides not to add the uninvited CU no
further action is required, however the "Organizer" MAY send the
uninvited CU a "CANCEL" message. If the "Organizer" decides to add
an uninvited CU, a new "ATTENDEE" property is added for the uninvited
CU with its property parameters set as the "Organizer" deems
appropriate. When forwarding a "REQUEST" to another CU, the
forwarding "Attendee" MUST NOT make changes to the VEVENT property
set.

3.2.2.7 Updating Attendee Status

The "Organizer" of an event may also request updated status from one
or more "Attendees. The "Organizer" sends a "REQUEST" method to the
"Attendee" and sets the "ATTENDEE;RSVP=TRUE" property parameter. 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 updated status. The recipient SHOULD respond with a "REPLY"
method indicating their current status with respect to the "REQUEST".

3.2.3 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 to provide a delegation response, the
"Delegator" SHOULD include the calendar address of the "Delegate" on
the "delegated-to" property parameter of the "Delegator's" "ATTENDEE"
property. The "Delegate" SHOULD include the calendar address of the
"Delegator" on the "delegated-from" property parameter of the
"Delegate'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 CU
not in the original "REQUEST". For example, a "REPLY" may be received
from a "Delegate" to an event. In addition, the "REPLY" method may be
received from an unknown CU (a "Party Crasher"). This uninvited
"Attendee" may be accepted, or the "Organizer" may cancel the event
for the uninvited "Attendee" by sending a "CANCEL" method to the
uninvited "Attendee".

An "Attendee" can include a message to the "Organizer" using the
"COMMENT" property. For example, if the user indicates tentative
acceptance and wants to let the "Organizer" know why, the reason can
be expressed in the "COMMENT" property value.

The "Organizer" may also receive a "REPLY" from one CU on behalf of
another. Like the scenario enumerated above for the "Organizer",
"Attendees" may have another CU respond on their behalf. This is done
using the "sent-by" parameter.

The optional properties listed in the table below (those listed as
"0+" or "0 or 1") MUST NOT be changed from those of the original
request. If property changes are desired the COUNTER message must be
used.

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

```
Component/Property  Presence
------------------- ----------------------------------------------
METHOD              1         MUST be "REPLY"
VEVENT              1+        All components MUST have the same UID
    ATTENDEE        1         MUST be the address of the Attendee
                              replying.
    DTSTAMP         1
    ORGANIZER       1
    RECURRENCE-ID   0 or 1    only if referring to an instance of a
                              recurring calendar component.  Otherwise
                              it must NOT be present.
    UID             1         MUST be the UID of the original REQUEST

    SEQUENCE        0 or 1    MUST if non-zero, MUST be the sequence
                              number of the original REQUEST. MAY be
                              present if 0.

    ATTACH          0+
    CATEGORIES      0 or 1    This property may contain a list of values
    CLASS           0 or 1
    COMMENT         0 or 1
    CONTACT         0+
    CREATED         0 or 1
    DESCRIPTION     0 or 1
    DTEND           0 or 1    if present DURATION MUST NOT be present
    DTSTART         0 or 1
    DURATION        0 or 1    if present DTEND MUST NOT be present
    EXDATE          0+
    EXRULE          0+
    GEO             0 or 1
    LAST-MODIFIED   0 or 1
    LOCATION        0 or 1
    PRIORITY        0 or 1
    RDATE           0+
    RELATED-TO      0+
```

```
        RESOURCES          0 or 1  This property MAY contain a list of values
        REQUEST-STATUS     0+
        RRULE              0+
        STATUS             0 or 1
        SUMMARY            0 or 1
        TRANSP             0 or 1
        URL                0 or 1
        X-PROPERTY         0+


VTIMEZONE             0 or 1 MUST be present if any date/time refers
                            to a timezone
X-COMPONENT           0+

VALARM                0
VFREEBUSY             0
VJOURNAL              0
VTODO                 0
```

3.2.4 ADD

   The "ADD" method in a "VEVENT" calendar component is used to add one
   or more instances to an existing "VEVENT". Unlike the "REQUEST"
   method, when using issuing an "ADD" method, the "Organizer" does not
   send the full "VEVENT" description; only the new instance(s). The
   "ADD" method is especially useful if there are instance-specific
   properties to be preserved in a recurring "VEVENT". For instance, an
   "Organizer" may have originally scheduled a weekly Thursday meeting.
   At some point, several instances changed. Location or start time may
   have changed, or some instances may have unique "DESCRIPTION"
   properties. The "ADD" method allows the "Organizer" to add new
   instances to an existing event using a single ITIP message without
   redefining the entire recurring pattern.

   The "UID" must be that of the existing event. If the "UID" property
   value in the "ADD" is not found on the recipient's calendar, then the
   recipient SHOULD send a "REFRESH" to the "Organizer" in order to be
   updated with the latest version of the "VEVENT". If an "Attendee"
   implementation does not support the "ADD" method it should respond
   with a "REQUEST-STATUS" value of 3.14 and ask for a "REFRESH".

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

```
Component/Property  Presence
------------------  -----------------------------------------------
METHOD              1       MUST be "ADD"
VEVENT              1
     DTSTAMP        1
     DTSTART        1
     ORGANIZER      1
     SEQUENCE       1       MUST be greater than 0
     SUMMARY        1       Can be null
     UID            1       MUST match that of the original event

     ATTACH         0+
     ATTENDEE       0+
     CATEGORIES     0 or 1 This property MAY contain a list of values
     CLASS          0 or 1
     COMMENT        0 or 1
     CONTACT        0+
     CREATED        0 or 1
     DESCRIPTION    0 or 1  Can be null
     DTEND          0 or 1  if present DURATION MUST NOT be present
     DURATION       0 or 1  if present DTEND MUST NOT be present
     EXDATE         0+
     EXRULE         0+
     GEO            0 or 1
     LAST-MODIFIED  0 or 1
     LOCATION       0 or 1
     PRIORITY       0 or 1
     RDATE          0+
     RELATED-TO     0+
     RESOURCES      0 or 1  This property MAY contain a list of values
     RRULE          0+
     STATUS         0 or 1  MAY be one of TENTATIVE/CONFIRMED
     TRANSP         0 or 1
     URL            0 or 1
     X-PROPERTY     0+

     RECURRENCE-ID  0
     REQUEST-STATUS 0

VALARM              0+
VTIMEZONE           0+      MUST be present if any date/time refers to
                           a timezone
X-COMPONENT         0+

VFREEBUSY           0
VTODO               0
VJOURNAL            0
```

3.2.5 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 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 and a
   "RECURRENCE-ID" MUST NOT 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.

   There are two options for canceling a sequence of instances of a
   recurring "VEVENT" calendar component:

   (a) the "RECURRENCE-ID" property for an instance in the sequence MUST
       be specified with the "RANGE" property parameter value of
       THISANDPRIOR (or THISANDFUTURE) to indicate cancellation of the
       specified "VEVENT" calendar component and all instances before
       (or after); or

   (b) individual recurrence instances may be cancelled by specifying
       multiple "RECURRENCE-ID" properties corresponding to the
       instances to be cancelled.

   When a "VEVENT" is cancelled, the "SEQUENCE" property value MUST be
   incremented.

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

Component/Property Presence
------------------ --
METHOD   1   MUST be "CANCEL"

VEVENT   1+   All must have the same UID
    ATTENDEE   0+   MUST include all "Attendees" being removed
                            the event. MUST include all "Attendees" if
                            the entire event is cancelled.
    DTSTAMP   1
    ORGANIZER   1
    SEQUENCE   1
    UID   1   MUST be the UID of the original REQUEST

    COMMENT   0 or 1
    ATTACH   0+
    CATEGORIES   0 or 1 This property may contain a list of values

```
      CLASS             0 or 1
      CONTACT           0+
      CREATED           0 or 1
      DESCRIPTION       0 or 1
      DTEND             0 or 1 if present DURATION MUST NOT be present
      DTSTART           0 or 1
      DURATION          0 or 1 if present DTEND MUST NOT be present
      EXDATE            0+
      EXRULE            0+
      GEO               0 or 1
      LAST-MODIFIED     0 or 1
      LOCATION          0 or 1
      PRIORITY          0 or 1
      RDATE             0+
      RECURRENCE-ID     0 or 1  MUST be present if referring to one or
                                more or more recurring instances.
                                Otherwise it MUST NOT be present
      RELATED-TO        0+
      RESOURCES         0 or 1
      RRULE             0+
      STATUS            0 or 1  MUST be set to CANCELLED. If uninviting
                                specific "Attendees" then MUST NOT be
                                included.
      SUMMARY           0 or 1
      TRANSP            0 or 1
      URL               0 or 1
      X-PROPERTY        0+
      REQUEST-STATUS    0

VTIMEZONE             0+      MUST be present if any date/time refers to
                             a timezone
X-COMPONENT           0+

VTODO                 0
VJOURNAL              0
VFREEBUSY             0
VALARM                0
```

3.2.6 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 of the event to update. A recurrence instance of an
   event may be requested by specifying the "RECURRENCE-ID" property
   corresponding to the associated event. The "Organizer" responds with
   the latest description and version of the event.

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

```
Component/Property   Presence
------------------   ----------------------------------------------
METHOD               1       MUST be "REFRESH"

VEVENT               1
    ATTENDEE         1       MUST be the address of requestor
    DTSTAMP          1
    ORGANIZER        1
    UID              1       MUST be the UID associated with original
                             REQUEST
    COMMENT          0 or 1
    RECURRENCE-ID    0 or 1 MUST only if referring to an instance of a
                             recurring calendar component.  Otherwise
                             it must NOT be present.
    X-PROPERTY       0+

    ATTACH           0
    CATEGORIES       0
    CLASS            0
    CONTACT          0
    CREATED          0
    DESCRIPTION      0
    DTEND            0
    DTSTART          0
    DURATION         0
    EXDATE           0
    EXRULE           0
    GEO              0
    LAST-MODIFIED    0
    LOCATION         0
    PRIORITY         0
    RDATE            0
    RELATED-TO       0
    REQUEST-STATUS   0
    RESOURCES        0
    RRULE            0
    SEQUENCE         0
    STATUS           0
    SUMMARY          0
    TRANSP           0
    URL              0

X-COMPONENT          0+

VTODO                0
```

```
VJOURNAL            0
VFREEBUSY           0
VTIMEZONE           0
VALARM              0
```

3.2.7 COUNTER

   The "COUNTER" method for 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" sends this
   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 "DECLINECOUNTER" method. The "Organizer" accepts
   the counter proposal by rescheduling the event as described in
   section 3.2.2.1 Rescheduling an Event.

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

```
Component/Property  Presence
------------------  ----------------------------------------------
METHOD              1       MUST be "COUNTER"

VEVENT              1
    DTSTAMP         1
    DTSTART         1
    ORGANIZER       1        MUST be the "Organizer" of the original
                             event
    SEQUENCE        1        MUST be present if value is greater than 0,
                             MAY be present if 0
    SUMMARY         1        Can be null
    UID             1        MUST be the UID associated with the REQUEST
                             being countered

    ATTACH          0+
    ATTENDEE        0+       Can also  be used to propose other
                             "Attendees"
    CATEGORIES      0 or 1  This property may contain a list of values
    CLASS           0 or 1
    COMMENT         0 or 1
    CONTACT         0+
    CREATED         0 or 1
    DESCRIPTION     0 or 1
```

```
      DTEND              0 or 1  if present DURATION MUST NOT be present
      DURATION           0 or 1  if present DTEND MUST NOT be present
      EXDATE             0+
      EXRULE             0+
      GEO                0 or 1
      LAST-MODIFIED      0 or 1
      LOCATION           0 or 1
      PRIORITY           0 or 1
      RDATE              0+
      RECURRENCE-ID      0 or 1  MUST only if referring to an instance of a
                                 recurring calendar component.  Otherwise it
                                 MUST NOT be present.
      RELATED-TO         0+
      REQUEST-STATUS     0+
      RESOURCES          0 or 1  This property may contain a list of values
      RRULE              0+
      STATUS             0 or 1  Value must be one of CONFIRMED/TENATIVE/
                                 CANCELLED
      TRANSP             0 or 1
      URL                0 or 1
      X-PROPERTY         0+

   VALARM                0+
   VTIMEZONE             0+      MUST be present if any date/time refers to
                                 a timezone
   X-COMPONENT           0+

   VTODO                 0
   VJOURNAL              0
   VFREEBUSY             0
```

3.2.8 DECLINECOUNTER

   The "DECLINECOUNTER" 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 "DECLINECOUNTER"
   message to the "Attendee" that sent the "COUNTER" method to the
   "Organizer".

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

```
Component/Property  Presence
------------------  ---------------------------------------------
METHOD              1       MUST be "DECLINECOUNTER"

VEVENT              1
    DTSTAMP         1
    ORGANIZER       1
    UID             1       MUST, same UID specified in original
                            REQUEST and subsequent COUNTER
    COMMENT         0 or 1
    RECURRENCE-ID   0 or 1  MUST only if referring to an instance of a
                            recurring calendar component.  Otherwise it
                            MUST NOT be present.
    REQUEST-STATUS  0+
    SEQUENCE        0 OR 1  MUST be present if value is greater than 0,
                            MAY be present if 0
    X-PROPERTY      0+
    ATTACH          0
    ATTENDEE        0
    CATEGORIES      0
    CLASS           0
    CONTACT         0
    CREATED         0
    DESCRIPTION     0
    DTEND           0
    DTSTART         0
    DURATION        0
    EXDATE          0
    EXRULE          0
    GEO             0
    LAST-MODIFIED   0
    LOCATION        0
    PRIORITY        0
    RDATE           0
    RELATED-TO      0
    RESOURCES       0
    RRULE           0
    STATUS          0
    SUMMARY         0
    TRANSP          0
    URL             0

X-COMPONENT         0+
VTODO               0
VJOURNAL            0
VFREEBUSY           0
VTIMEZONE           0
VALARM              0
```

3.3 Methods For VFREEBUSY Components

   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 restriction table.

   This document 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 ([US-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 individual "D" returns an
   "unsupported capability" message to individual "A4.34.3".

   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.                  |
|===============|================================================|
```

3.3.1 PUBLISH

   The "PUBLISH" method in a "VFREEBUSY" calendar component is used to
   publish busy time data. The method may be sent from one CU to any
   other. The purpose of the method is to provide a 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.

   The "ATTENDEE" property must be specified in the busy time
   information. The value is the CU address of the originator of the
   busy time information.

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

```
Component/Property  Presence
------------------- ----------------------------------------------
METHOD              1       MUST be "PUBLISH"

VFREEBUSY           1+
    DTSTAMP         1
    DTSTART         1       DateTime values must be in UTC
    DTEND           1       DateTime values must be in UTC
    FREEBUSY        1+      MUST be BUSYTIME. Multiple instances are
                            allowed. Multiple instances must be sorted
                            in ascending order
    ORGANIZER       1       MUST contain the address of originator of
                            busy time data.

    COMMENT         0 or 1
    CONTACT         0+
    X-PROPERTY      0+
    URL             0 or 1  Specifies busy time URL

    ATTENDEE        0
    DURATION        0
    REQUEST-STATUS  0
    UID             0

X-COMPONENT         0+
```

```
VEVENT                 0
VTODO                  0
VJOURNAL               0
VTIMEZONE              0
VALARM                 0
```

3.3.2 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.

   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 SHOULD be 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:

```
Component/Property  Presence
------------------- ------------------------------------------------
METHOD              1      MUST be "REQUEST"

VFREEBUSY           1
    ATTENDEE        1+     contain the address of the calendar store
    DTEND           1      DateTime values must be in UTC
    DTSTAMP         1
    DTSTART         1      DateTime values must be in UTC
    ORGANIZER       1      MUST be the request originator's address
    UID             1
    COMMENT         0 or 1
    CONTACT         0+
    X-PROPERTY      0+

    FREEBUSY        0
    DURATION        0
    REQUEST-STATUS  0
    URL             0

X-COMPONENT         0+
VALARM              0
VEVENT              0
```

```
VTODO               0
VJOURNAL            0
VTIMEZONE           0
```

3.3.3 REPLY

   The "REPLY" method in a "VFREEBUSY" calendar component is used to
   respond to a busy time request. The method is sent by the recipient
   of a busy time request to the originator of the request.

   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.

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

```
Component/Property  Presence
------------------  ---------------------------------------------
METHOD              1       MUST be "REPLY"

VFREEBUSY           1
    ATTENDEE        1       (address of recipient replying)
    DTSTAMP         1
    DTEND           1       DateTime values must be in UTC
    DTSTART         1       DateTime values must be in UTC
    FREEBUSY        1+       (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)
    ORGANIZER       1       MUST be the request originator's address
    UID             1

    COMMENT         0 or 1
    CONTACT         0+
    REQUEST-STATUS  0+
    URL             0 or 1  (specifies busy time URL)
    X-PROPERTY      0+
    DURATION        0
    SEQUENCE        0

X-COMPONENT         0+
VALARM              0
VEVENT              0
VTODO               0
VJOURNAL            0
VTIMEZONE           0
```

3.4 Methods For VTODO Components

   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 restriction table 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. The REQUEST method     |
   |                | is also used to update or change an existing       |
   |                | VTODO. Clients that cannot handle REQUEST MAY      |
   |                | degrade the method to treat it as a PUBLISH.       |
   |                |                                                    |
   | REPLY          | Reply to a VTODO request. Attendees MAY set        |
   |                | PARTSTAT to ACCEPTED, DECLINED, TENTATIVE,         |
   |                | DELEGATED, PARTIAL, and COMPLETED.                 |
   |                |                                                    |
   | ADD            | Add one or more instances to an existing to-do.    |
   |                |                                                    |
   | CANCEL         | Cancel one or more instances of an existing        |
   |                | to-do.                                             |
   |                |                                                    |
   | REFRESH        | A request sent to a VTODO Organizer asking for     |
   |                | the latest version of a VTODO.                     |
   |                |                                                    |
   | COUNTER        | Counter a REQUEST with an alternative proposal.    |
   |                |                                                    |
   | DECLINECOUNTER | Decline a counter proposal by an Attendee.         |
   +================+====================================================+
```

3.4.1 PUBLISH

   The "PUBLISH" method in a "VTODO" calendar component has no
   associated response. It is simply a posting of an iCalendar object
   that maybe added to a calendar. It MUST have an "Organizer". It MUST
   NOT have "Attendees". Its expected usage is for encapsulating an
   arbitrary "VTODO" calendar component as an iCalendar object. The
   "Organizer" MAY subsequently update (with another "PUBLISH" method),
   add instances to (with an "ADD" method), or cancel (with a "CANCEL"

method) a previously published "VTODO" calendar component.

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

```
Component/Property  Presence
------------------  ----------------------------------------------
METHOD              1         MUST be "PUBLISH"
VTODO               1+
    DTSTAMP         1
    DTSTART         1
    ORGANIZER       1
    PRIORITY        1
    SEQUENCE        0 or 1  MUST be present if value is greater than
                            0, MAY be present if 0
    SUMMARY         1       Can be null.
    UID             1

    ATTACH          0+
    CATEGORIES      0 or 1  This property may contain a list of values
    CLASS           0 or 1
    COMMENT         0 or 1
    CONTACT         0+
    CREATED         0 or 1
    DESCRIPTION     0 or 1  Can be null
    DUE             0 or 1  If present DURATION MUST NOT be present
    DURATION        0 or 1  If present DUE MUST NOT be present
    EXDATE          0+
    EXRULE          0+
    GEO             0 or 1
    LAST-MODIFIED   0 or 1
    LOCATION        0 or 1
    PERCENT-COMPLETE 0 or 1
    RDATE           0+
    RECURRENCE-ID   0 or 1  MUST only if referring to an instance of a
                            recurring calendar component.  Otherwise
                            it MUST NOT be present.

    RELATED-TO      0+
    RESOURCES       0 or 1  This property may contain a list of values
    RRULE           0+
STATUS              0 or 1  MAY be one of COMPLETED/NEEDS ACTION/IN-
                            PROCESS/CANCELLED
    URL             0 or 1
    X-PROPERTY      0+

    ATTENDEE        0
    REQUEST-STATUS  0
```

```
VTIMEZONE              0+      MUST be present if any date/time refers to
                               a timezone
VALARM                 0+
X-COMPONENT            0+

VFREEBUSY              0
VEVENT                 0
VJOURNAL               0
```

3.4.2 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=REQ-PARTICIPANT" 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.
   The "Attendee" uses the "REPLY" method to convey their acceptance and
   completion status to the "Organizer" of the "REQUEST".

   The "UID", "SEQUENCE", and "DTSTAMP" 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 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.

   For the "REQUEST" method, multiple "VTODO" components in a single
   iCalendar object are only permitted when for components with the same
   "UID" property. That is, a series of recurring events may have

instance-specific information. In this case, multiple "VTODO"
components are needed to express the entire series.

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

```
Component/Property  Presence
------------------  ------------------------------------------------
METHOD              1         MUST be "REQUEST"
VTODO               1+        All components must have the same UID
    ATTENDEE        1+
    DTSTAMP         1
    DTSTART         1
    ORGANIZER       1
    PRIORITY        1
    SEQUENCE        0 or 1  MUST be present if value is greater than
                            0, MAY be present if 0
    SUMMARY         1       Can be null.
    UID             1

    ATTACH          0+
    CATEGORIES      0 or 1   This property may contain a list of
                             values
    CLASS           0 or 1
    COMMENT         0 or 1
    CONTACT         0+
    CREATED         0 or 1
    DESCRIPTION     0 or 1  Can be null
    DUE             0 or 1  If present DURATION MUST NOT be present
    DURATION        0 or 1  If present DUE MUST NOT be present
    EXDATE          0+
    EXRULE          0+
    GEO             0 or 1
    LAST-MODIFIED   0 or 1
    LOCATION        0 or 1
    PERCENT-COMPLETE 0 or 1
    RDATE           0+
    RECURRENCE-ID   0 or 1  present if referring to an instance of a
                            recurring calendar component.  Otherwise
                            it MUST NOT be present.
    RELATED-TO      0+
    RESOURCES       0 or 1  This property may contain a list of
                            values
    RRULE           0+
    STATUS          0 or 1  MAY be one of COMPLETED/NEEDS ACTION/IN-
                            PROCESS
    URL             0 or 1
    X-PROPERTY      0+
```

```
   REQUEST-STATUS      0

VALARM                 0+

VTIMEZONE              0+  MUST be present if any date/time refers
                             to a timezone
X-COMPONENT            0+

VEVENT                 0
VFREEBUSY              0
VJOURNAL               0
```

3.4.2.1 REQUEST for Rescheduling a VTODO

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

   Rescheduling a "VTODO" calendar component involves a change to the
   existing "VTODO" calendar component in terms of its 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.

3.4.2.2 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 not a 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 be sent by the "Organizer" of a
   "VTODO" calendar component. The unsolicited "REQUEST" methods are

   used to update the details of the "VTODO" (without rescheduling it or
   updating the completion status of "Attendees") or the "VTODO"
   calendar component itself (i.e., reconfirm the "VTODO").

3.4.2.3 REQUEST for Delegating a VTODO

   The "REQUEST" method is also used to delegate or reassign ownership
   of a "VTODO" calendar component to another "Calendar User". For
   example, it may be used to delegate an "Attendee's" role (i.e.
   "chair", or "participant") for a "VTODO" calendar component. The
   "REQUEST" method is sent by one of the "Attendees" of an existing

   "VTODO" calendar component to some other individual. An "Attendee" of
   a "VTODO" calendar component MUST 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
   "Delegate".

   The "Delegator" of a "VTODO" calendar component MUST forward the
   existing "REQUEST" method for a "VTODO" calendar component to the
   "Delegate". 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 "Delegate". The "Delegator" MUST also
   send a "REPLY" method back to the "Organizer" with the "Delegator's"
   "Attendee" property "partstat" parameter value set to "DELEGATED". In
   addition, the "delegated-to" parameter MUST be included with the
   calendar address of the "Delegate". A response to the delegation
   "REQUEST" is sent from the "Delegate" to the "Organizer" and
   optionally, to the "Delegator". The "REPLY" method from the
   "Delegate" 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 a value for the "RSVP"
   property parameter associated with the "Delegator's" "Attendee"
   property to that of the "Delegate's" "ATTENDEE" property. For example
   if the "Delegator's" "ATTENDEE" property specifies "RSVP=TRUE", then
   the "Delegate's" "ATTENDEE" property MUST specify "RSVP=TRUE".

3.4.2.4 REQUEST Forwarded To An Uninvited Calendar User

   An "Attendee" assigned a "VTODO" calendar component may send the
   "VTODO" calendar component to another new CU, not previously
   associated with the "VTODO" calendar component. The current

"Attendee" assigned the "VTODO" calendar component does this by
forwarding the original "REQUEST" method to the new CU. The new CU
can send a "REPLY" to the "Organizer" of the "VTODO" calendar
component. The reply contains an "ATTENDEE" property for the new CU.

The "Organizer" ultimately decides whether or not the new CU becomes
part of the to-do and is not obligated to do anything with a "REPLY"
from a new (uninvited) CU. If the "Organizer" does not want the new
CU to be part of the to-do, the new "ATTENDEE" property is not added
to the "VTODO" calendar component. The "Organizer" MAY send the CU a
"CANCEL" message to indicate that they will not be added to the to-
do. If the "Organizer" decides to add the new CU, the new "ATTENDEE"
property is added to the "VTODO" calendar component. Furthermore, the
"Organizer" is free to change any "ATTENDEE" property parameter from
the values supplied by the new CU to something the "Organizer"
considers appropriate.

3.4.2.5 REQUEST Updated Attendee Status

An "Organizer" of a "VTODO" may request an updated status from one or
more "Attendees". The "Organizer" sends a "REQUEST" method to the
"Attendee" with the "ATTENDEE;RSVP=TRUE" property sequence. The
"SEQUENCE" property for the "VTODO" is not changed from its previous
value. A recipient determines 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".

3.4.3 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"
MUST include the calendar address of the "Delegate" in the
"delegated-to" parameter of the "Delegator's" "ATTENDEE" property.
The "Delegate" MUST include the calendar address of the "Delegator"
on the "delegated-from" parameter of the "Delegate'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 "Delegate" of a
"VTODO" calendar component. In addition, the "REPLY" method MAY be

   received from an unknown "Calendar User", having been forwarded the
   "REQUEST" by an original "Attendee" of 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:

```
Component/Property   Presence
------------------   ---------------------------------------------
METHOD               1       MUST be "REPLY"
VTODO                1+      All component MUST have the same UID
    ATTENDEE         1+
    DTSTAMP          1
    ORGANIZER        1
    REQUEST-STATUS   1+
    UID              1       MUST must be the address of the replying
                             attendee

    ATTACH           0+
    CATEGORIES       0 or 1 This property may contain a list of values
    CLASS            0 or 1
    COMMENT          0 or 1
    CONTACT          0+
    CREATED          0 or 1
    DESCRIPTION      0 or 1
    DTSTART          0 or 1
    DUE              0 or 1  If present DURATION MUST NOT be present
    DURATION         0 or 1  If present DUE MUST NOT be present
    EXDATE           0+
    EXRULE           0+
    GEO              0 or 1
    LAST-MODIFIED    0 or 1
    LOCATION         0 or 1
    PERCENT-COMPLETE 0 or 1
    PRIORITY         0 or 1
    RDATE            0+
    RELATED-TO       0+
    RESOURCES        0 or 1  This property may contain a list of values
    RRULE            0+
    RECURRENCE-ID    0 or 1  MUST only if referring to an instance of a
                             Recurring calendar component. Otherwise it
                             MUST NOT be present
    SEQUENCE         0 or 1  MUST be the sequence number of
                             the original REQUEST if greater than 0.
                             MAY be present if 0.
    STATUS           0 or 1
```

```
    SUMMARY              0 or 1  Can be null
    URL                  0 or 1
    X-PROPERTY           0+

VTIMEZONE                0 or 1  MUST be present if any date/time refers to
                                 a timezone
X-COMPONENT              0+

VALARM                   0
VEVENT                   0
VFREEBUSY                0
```

3.4.4 ADD

   The "ADD" method in a "VTODO" calendar component is used to add one
   or more instances to an existing to-do.

   If the "UID" property value in the "ADD" is not found on the
   recipient's calendar, then the recipient SHOULD send a "REFRESH" to
   the "Organizer" in order to be updated with the latest version of the
   "VTODO". If an "Attendee" implementation does not support the "ADD"
   method it should respond with a "REQUEST-STATUS" value of 5.3 and ask
   for a "REFRESH".

   The "SEQUENCE" property value is incremented as the sequence of to-
   dos has changed.

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

```
Component/Property  Presence
------------------  ----------------------------------------------
METHOD                   1       MUST be "ADD"
VTODO                    1
    DTSTAMP              1
    ORGANIZER            1
    PRIORITY             1
    SEQUENCE             1       MUST be greater than 0
    SUMMARY              1       Can be null.
    UID                  1       MUST match that of the original to-do

    ATTACH               0+
    ATTENDEE             0+
    CATEGORIES           0 or 1  This property may contain a list of
                                 values
    CLASS                0 or 1
    COMMENT              0 or 1
    CONTACT              0+
```

```
   CREATED              0 or 1
   DESCRIPTION          0 or 1  Can be null
   DTSTART              0 or 1
   DUE                  0 or 1  If present DURATION MUST NOT be present
   DURATION             0 or 1  If present DUE MUST NOT be present
   EXDATE               0+
   EXRULE               0+
   GEO                  0 or 1
   LAST-MODIFIED        0 or 1
   LOCATION             0 or 1
   PERCENT-COMPLETE     0 or 1
   RDATE                0+
   RELATED-TO           0+
   RESOURCES            0 or 1  This property may contain a list of
                                values
   RRULE                0+
   STATUS               0 or 1  MAY be one of COMPLETED/NEEDS ACTION/IN-
                                PROCESS
   URL                  0 or 1
   X-PROPERTY           0+

   RECURRENCE-ID        0
   REQUEST-STATUS       0

VALARM                  0+
VTIMEZONE               0+       MUST be present if any date/time refers
                                 to a timezone
X-COMPONENT             0+

VEVENT                  0
VJOURNAL                0
VFREEBUSY               0
```

3.4.5 CANCEL

   The "CANCEL" method in a "VTODO" calendar component is used to send a
   cancellation notice of an existing "VTODO" calendar 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 and a "RECURRENCE-
   ID" MUST NOT 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.

There are two options for canceling a sequence of instances of a
recurring "VTODO" calendar component:

(a) the "RECURRENCE-ID" property for an instance in the sequence MUST
    be specified with the "RANGE" property parameter value of
    THISANDPRIOR (or THISANDFUTURE) to indicate cancellation of the
    specified "VTODO" calendar component and all instances before (or
    after); or

(b) individual recurrence instances may be cancelled by specifying
    multiple "RECURRENCE-ID" properties corresponding to the
    instances to be cancelled.

When a "VTODO" is cancelled, the "SEQUENCE" property value MUST be
incremented.

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

```
Component/Property    Presence
------------------    --------------------------------------------
METHOD                1     MUST be "CANCEL"
VTODO                 1
    ATTENDEE          0+    include all "Attendees" being removed from
                           the todo. MUST include all "Attendees" if
                           the entire todo is cancelled.
    UID               1     MUST echo original UID
    DTSTAMP           1
    ORGANIZER         1
    SEQUENCE          1

    ATTACH            0+
    CATEGORIES        0 or 1 This property MAY contain a list of values
    CLASS             0 or 1
    COMMENT           0 or 1
    CONTACT           0+
    CREATED           0 or 1
    DESCRIPTION       0 or 1
    DTSTART           0 or 1
    DUE               0 or 1  If present DURATION MUST NOT be present
    DURATION          0 or 1  If present DUE MUST NOT be present
    EXDATE            0+
    EXRULE            0+
    GEO               0 or 1
    LAST-MODIFIED     0 or 1
    LOCATION          0 or 1
    PERCENT-COMPLETE  0 or 1
    RDATE             0+
```

```
        RECURRENCE-ID    0 or 1 MUST only if referring to one or more
                                instances of a recurring calendar
                                component. Otherwise it MUST NOT be
                                present.
        RELATED-TO       0+
        RESOURCES        0 or 1 This property MAY contain a list of values
        RRULE            0+
        PRIORITY         0 or 1
        STATUS           0 or 1  If present it MUST be set to "CANCELLED".
                                 MUST NOT be used if purpose is to remove
                                 "ATTENDEES" rather than cancel the entire
                                 VTODO.
        URL              0 or 1
        X-PROPERTY       0+

        REQUEST-STATUS   0

VTIMEZONE                0 or 1  MUST be present if any date/time refers to
                                 a timezone
X-COMPONENT              0+

VALARM                   0
VEVENT                   0
VFREEBUSY                0
```

3.4.6 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 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" responds with the latest description and rendition of the
   "VTODO" calendar component. In most cases this will be a REQUEST
   unless the "VTODO" has been cancelled, in which case the ORGANIZER
   MUST send a "CANCEL". 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:

```
Component/Property   Presence
------------------   ----------------------------------------------
METHOD               1        MUST be "REFRESH"
VTODO                1
    ATTENDEE         1
    DTSTAMP          1
    UID              1        MUST echo original UID

    RECURRENCE-ID    0 or 1   MUST only if referring to an instance of a
                              Recurring calendar component. Otherwise it
                              MUST NOT be present
    X-PROPERTY       0+

    ATTACH           0
    CATEGORIES       0
    CLASS            0
    COMMENT          0
    CONTACT          0
    CREATED          0
    DESCRIPTION      0
    DTSTART          0
    DUE              0
    DURATION         0
    EXDATE           0
    EXRULE           0
    GEO              0
    LAST-MODIFIED    0
    LOCATION         0
    ORGANIZER        0
    PERCENT-COMPLETE 0
    PRIORITY         0
    RDATE            0
    RELATED-TO       0
    REQUEST-STATUS   0
    RESOURCES        0
    RRULE            0
    SEQUENCE         0
    STATUS           0
    URL              0

X-COMPONENT          0+

VALARM               0
VEVENT               0
VFREEBUSY            0
VTIMEZONE            0
```

3.4.7 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" sends 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 "DECLINECOUNTER" 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 latter 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:

```
Component/Property  Presence
------------------  ----------------------------------------------
METHOD              1       MUST be "COUNTER"
VTODO               1
    ATTENDEE        1+
    DTSTAMP         1
    ORGANIZER       1
    PRIORITY        1
    SUMMARY         1      Can be null
    UID             1

    ATTACH          0+
    CATEGORIES      0 or 1 This property MAY contain a list of values
    CLASS           0 or 1
    COMMENT         0 or 1
    CONTACT         0+
    CREATED         0 or 1
    DESCRIPTION     0 or 1 Can be null
    DTSTART         0 or 1
    DUE             0 or 1  If present DURATION MUST NOT be present
    DURATION        0 or 1  If present DUE MUST NOT be present
    EXDATE          0+
    EXRULE          0+
    GEO             0 or 1
    LAST-MODIFIED   0 or 1
```

```
        LOCATION           0 or 1
        PERCENT-COMPLETE 0 or 1
        RDATE              0+
        RECURRENCE-ID      0 or 1 MUST only 3.5if referring to an instance of a
                                  recurring calendar component.  Otherwise it
                                  MUST NOT be present.
        RELATED-TO         0+
        REQUEST-STATUS     0+
        RESOURCES          0 or 1 This property MAY contain a list of values
        RRULE              0 or 1
        SEQUENCE           0 or 1 MUST echo the original SEQUENCE number.
                                  MUST be present if non-zero. MAY be present
                                  if zero.
        STATUS             0 or 1 MAY be one of COMPLETED/NEEDS ACTION/IN-
                                  PROCESS/CANCELLED
        URL                0 or 1
        X-PROPERTY         0+


   VALARM                  0+
   VTIMEZONE               0 or 1  MUST be present if any date/time refers to
                                    a timezone
   X-COMPONENT             0+

   VEVENT                  0
   VFREEBUSY               0
```

3.4.8 DECLINECOUNTER

   The "DECLINECOUNTER" 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" sends the
   message to the "Attendee" that sent the "COUNTER" method to the
   "Organizer".

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

```
Component/Property   Presence
------------------   --------------------------------------------
METHOD               1       MUST be "DECLINECOUNTER"

VTODO                1
    ATTENDEE         1+      MUST for all attendees
    DTSTAMP          1
    ORGANIZER        1
    SEQUENCE         1       MUST echo the original SEQUENCE number
    UID              1       MUST echo original UID
```

```
     ATTACH              0+
     CATEGORIES          0 or 1  This property may contain a list of values
     CLASS               0 or 1
     COMMENT             0 or 1
     CONTACT             0+
     CREATED             0 or 1
     DESCRIPTION         0 or 1
     DTSTART             0 or 1
     DUE                 0 or 1  If present DURATION MUST NOT be present
     DURATION            0 or 1  If present DUE MUST NOT be present
     EXDATE              0+
     EXRULE              0+
     GEO                 0 or 1
     LAST-MODIFIED       0 or 1
     LOCATION            0 or 1
     PERCENT-COMPLETE    0 or 1
     PRIORITY            0 or 1
     RDATE               0+
     RECURRENCE-ID       0 or 1  MUST only if referring to an instance of a
                                 recurring calendar component.  Otherwise
                                 it MUST NOT be present.
     RELATED-TO          0+
     REQUEST-STATUS      0+
     RESOURCES           0 or 1  This property MAY contain a list of values
     RRULE               0+
     STATUS              0 or 1  MAY be one of COMPLETED/NEEDS ACTION/IN-
                                 PROCESS
     URL                 0 or 1
     X-PROPERTY          0+

 VTIMEZONE               0+  MUST be present if any date/time refers to
                             a timezone
 X-COMPONENT             0+

 VALARM                  0
 VEVENT                  0
 VFREEBUSY               0
```

3.5 Methods For VJOURNAL Components

   This section defines the property set for the methods that are
   applicable to the "VJOURNAL" calendar component.

   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. |
|               |                                                  |
| ADD           | Add one or more instances to an existing journal |
|               | entry.                                           |
|               |                                                  |
| CANCEL        | Cancel one or more instances of an existing      |
|               | journal entry.                                   |
+===============+==================================================+
```

3.5.1 PUBLISH

   The "PUBLISH" method in a "VJOURNAL" calendar component has no
   associated response. It is simply a posting of an iCalendar object
   that may be added to a calendar. It MUST have an "Organizer". It MUST
   NOT have "Attendees". 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:

```
Component/Property  Presence
------------------  -----------------------------------------------
METHOD              1        MUST be "PUBLISH"
VJOURNAL            1+
    DESCRIPTION     1        Can be null.
    DTSTAMP         1
    DTSTART         1
    ORGANIZER       1
    UID             1

    ATTACH          0+
    CATEGORIES      0 or 1   This property MAY contain a list of values
    CLASS           0 or 1
    COMMENT         0 or 1
    CONTACT         0+
    CREATED         0 or 1
    EXDATE          0+
    EXRULE          0+
    LAST-MODIFIED   0 or 1
    RDATE           0+
    RECURRENCE-ID   0 or 1   MUST only if referring to an instance of a
```

```
                                recurring calendar component.  Otherwise
                                it MUST NOT be present.
       RELATED-TO      0+
       RRULE           0+
       SEQUENCE        0 or 1  MUST echo the original SEQUENCE number.
                                MUST be present if non-zero. MAY be
                                present if zero.
       STATUS          0 or 1  MAY be one of DRAFT/FINAL/CANCELLED
       SUMMARY         0 or 1  Can be null
       URL             0 or 1
       X-PROPERTY      0+

       ATTENDEE        0

  VALARM              0+
  VTIMEZONE           0+       MUST be present if any date/time refers to
                                a timezone
  X-COMPONENT         0+

  VEVENT              0
  VFREEBUSY           0
  VTODO               0
```

3.5.2 ADD

   The "ADD" method in a "VJOURNAL" calendar component is used to add
   one or more instances to an existing "VJOURNAL" entry. There is no
   response to the "Organizer".

   If the "UID" property value in the "ADD" is not found on the
   recipient's calendar, then the recipient MAY treat the "ADD" as a
   "PUBLISH".

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

```
Component/Property  Presence
------------------- ----------------------------------------------
METHOD              1         MUST be "ADD"
VJOURNAL            1
    DESCRIPTION     1         Can be null.
    DTSTAMP         1
    DTSTART         1
    ORGANIZER       1
    SEQUENCE        1         MUST be greater than 0
    UID             1         MUST match that of the original journal

    ATTACH          0+
```

```
       CATEGORIES         0 or 1 This property MAY contain a list of values
       CLASS              0 or 1
       COMMENT            0 or 1
       CONTACT            0+
       CREATED            0 or 1
       EXDATE             0+
       EXRULE             0+
       LAST-MODIFIED      0 or 1
       RDATE              0+
       RELATED-TO         0+
       RRULE              0+
       STATUS             0 or 1  MAY be one of DRAFT/FINAL/CANCELLED
       SUMMARY            0 or 1  Can be null
       URL                0 or 1
       X-PROPERTY         0+

       ATTENDEE           0
       RECURRENCE-ID      0

VALARM                    0+
VTIMEZONE                 0 or 1 MUST be present if any date/time refers to
                                 a timezone
X-COMPONENT               0+

VEVENT                    0
VFREEBUSY                 0
VTODO                     0
```

3.5.3 CANCEL

   The "CANCEL" method in a "VJOURNAL" calendar component is used to
   send a cancellation notice of an existing journal entry. The message
   is sent by the "Organizer" of a 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 and a "RECURRENCE-ID" property MUST NOT 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.

   There are two options for canceling a sequence of instances of a
   recurring "VJOURNAL" calendar component:

(a) the "RECURRENCE-ID" property for an instance in the sequence MUST
    be specified with the "RANGE" property parameter value of
    THISANDPRIOR (or THISANDFUTURE) to indicate cancellation of the
    specified "VTODO" calendar component and all instances before (or
    after); or

(b) individual recurrence instances may be cancelled by specifying
    multiple "RECURRENCE-ID" properties corresponding to the
    instances to be cancelled.

When a "VJOURNAL" is cancelled, the "SEQUENCE" property value MUST be
incremented.

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

```
Component/Property    Presence
------------------    ---------------------------------------------
METHOD                1         MUST be "CANCEL"
VJOURNAL              1+        All MUST have the same UID
    DTSTAMP           1
    ORGANIZER         1
    SEQUENCE          1
    UID               1         MUST be the UID of the original REQUEST

    ATTACH            0+
    ATTENDEE          0+
    CATEGORIES        0 or 1    This property MAY contain a list of values
    CLASS             0 or 1
    COMMENT           0 or 1
    CONTACT           0+
    CREATED           0 or 1
    DESCRIPTION       0 or 1
    DTSTART           0 or 1
    EXDATE            0+
    EXRULE            0+
    LAST-MODIFIED     0 or 1
    RDATE             0+
    RECURRENCE-ID     0 or 1    only if referring to an instance of a
                                recurring calendar component.  Otherwise
                                it MUST NOT be present.
    RELATED-TO        0+
    RRULE             0+
    STATUS            0 or 1    MAY be present, must be "CANCELLED" if
                                present
    SUMMARY           0 or 1
    URL               0 or 1
    X-PROPERTY        0+
```

```
       REQUEST-STATUS    0

VTIMEZONE              0+       MUST be present if any date/time refers to
                                a timezone
X-COMPONENT            0+
VALARM                 0
VEVENT                 0
VFREEBUSY              0
VTODO                  0
```

3.6 Status Replies

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

Short Return Status Code	Longer Return Status Description	Offending Data
==========================	==================================	================
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	RRULE or RDATE property

	ignored. Scheduled as a single component.	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.
2.11	Success, unbounded RRULE clipped at some finite number of instances	RRULE property name and value MAY be specified. Number of instances MAY also 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.	METHOD 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.
======	======================	===
3.11	Required component or property missing.	Component or property name MAY be specified.
======	======================	===
3.12	Unknown component or property found	Component or property name MAY be specified
======	======================	===
3.13	Unsupported component or property found	Component or property name MAY be specified
======	======================	===
3.14	Unsupported capability	Method or action MAY be specified
======	======================	===
4.0	Event conflict. Date/time is busy.	DTSTART and DTEND property name and values MAY be specified.
======	======================	===
5.0	Request MAY 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.
======	======================	===

3.7 Implementation Considerations

3.7.1 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 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
"partstat" 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
effect 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 is renegotiated.
Negotiation may be necessary when a substantive change to the event
or to-do has be made (such as changing the start time, end time, due
date or location). The "Organizer" can identify a specific recurrence
instance 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. Note that after the change has occurred, the
"RECURRENCE-ID" has changed to the new "DTSTART" value.

3.7.2 Attendee Property Considerations

The "ORGANIZER" property is required on published events, to-dos, and
journal entries for two reasons. First, only the "Organizer" is
allowed to update and redistribute an event or to-do component. It
follows that the "ORGANIZER" 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.

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 then determines if the "Calendar User"
     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
       . the "member" 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.

3.7.3 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 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 are 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
ORGANIZER:mailto:a@example.com
DTSTART:19970701T200000Z
DTSTAMP:19970611T190000Z
SUMMARY:ST. PAUL SAINTS -VS- DULUTH-SUPERIOR DUKES
UID:0981234-1234234-23@example.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
ORGANIZER:mailto:a@example.com
DTSTAMP:19970612T190000Z
DTSTART:19970701T210000Z
DTEND:19970701T230000Z
SEQUENCE:1
UID:0981234-1234234-23@example.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 a change to the event. The
event with a matching UID and sequence number 0 is 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 MUST 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
ORGANIZER:mailto:a@example.com
COMMENT:DUKES forfeit the game
SEQUENCE:2
UID:0981234-1234234-23@example.com
DTSTAMP:19970613T190000Z
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
   VERSION:2.0
   BEGIN:VTIMEZONE
   TZID:America-Chicago
   TZURL:http://zones.stds_r_us.net/tz/America-Chicago
   BEGIN:STANDARD
   DTSTART:19671029T020000
   RRULE:FREQ=YEARLY;BYDAY=-1SU;BYMONTH=10
   TZOFFSETFROM:-0500
   TZOFFSETTO:-0600
   TZNAME:CST
   END:STANDARD
   BEGIN:DAYLIGHT
   DTSTART:19870405T020000
   RRULE:FREQ=YEARLY;BYDAY=1SU;BYMONTH=4
   TZOFFSETFROM:-0600
   TZOFFSETTO:-0500
   TZNAME:CDT
   END:DAYLIGHT
   END:VTIMEZONE
   BEGIN:VEVENT
   ORGANIZER:mailto:a@example.com
   ATTACH:http://www.dukes.com/
   CATEGORIES:SPORTS EVENT,ENTERTAINMENT
   CLASS:PRIVATE
   DESCRIPTION:MIDWAY STADIUM\n
    Big time game. MUST see.\n
    Expected duration:2 hours\n
   DTEND;TZID=America-Chicago:19970701T180000
   DTSTART;TZID=America-Chicago:19970702T160000
   DTSTAMP:19970614T190000Z
   STATUS:CONFIRMED
   LOCATION;VALUE=URI:http://www.midwaystadium.com/
   PRIORITY:2
   RESOURCES:SCOREBOARD
   SEQUENCE:3
   SUMMARY:ST. PAUL SAINTS -VS- DULUTH-SUPERIOR DUKES
   UID:0981234-1234234-23@example.com
   RELATED-TO:0981234-1234234-14@example.com
   BEGIN:VALARM

```
TRIGGER:-PT2H
ACTION:DISPLAY
DESCRIPTION:You should be leaving for the game now.
END:VALARM
BEGIN:VALARM
TRIGGER:-PT30M
ACTION:AUDIO
END:VALARM
END:VEVENT
END:VCALENDAR
```

The "RELATED-TO" field contains the "UID" property of a related
calendar event. 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
ORGANIZER:mailto:a@example.com
DTSTAMP:19970614T190000Z
UID:0981234-1234234-23@example.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" and the "Organizer" 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 attendance.
   The following table illustrates the the message flow between these
   individuals. A, the CU scheduling the meeting, is referenced as the
   "Organizer".

```
+----------------------------------------------------------------+
| Action              | "Organizer"        | Attendee            |
+----------------------------------------------------------------+
| Initiate a meeting  | "A" sends a REQUEST |                     |
| request             | message to "B", "C",|                     |
|                     | and "D"            |                     |
+----------------------------------------------------------------+
| Accept the meeting  |                    | "B" sends a REPLY    |
| request             |                    | message to "A" with its|
|                     |                    | ATTENDEE "partstat" para-|
|                     |                    | set to "accepted"   |
+----------------------------------------------------------------+
| Decline the meeting |                    | "C" sends a REPLY    |
| request             |                    | message to "A" with its|
|                     |                    | ATTENDEE "partstat" para-|
|                     |                    | set to "declined"   |
+----------------------------------------------------------------+
| Tentatively accept  |                    | "D" sends a REPLY    |
| the meeting request |                    | message to "A" with its|
|                     |                    | ATTENDEE "partstat" para-|
|                     |                    | set to "tentative"  |
+----------------------------------------------------------------+
| Confirm meeting     | "A" sends a REQUEST |                     |
| status with         | message to "B" and |                     |
| attendees           | "D" with updated   |                     |
|                     | information.       |                     |
+----------------------------------------------------------------+
```

4.2.1 A Group Event Request

   A sample meeting request is sent from "A" to "B", "C", and "D". _E_
   is also sent a copy of the request but is not expected to attend and
   need not reply. "E" illustrates how CUAs might implement an "FYI"
   type feature. Note the use of the "role" parameter. The default value
   for the "role" parameter is "req-participant" and it need not be
   enumerated. In this case we are using the value "non-participant" to
   indicate "E" is a non-attending CU. The parameter is not needed on
   other "Attendees" since "participant" is the default value.

   BEGIN:VCALENDAR
   PRODID:-//ACME/DesktopCalendar//EN
   METHOD:REQUEST
   VERSION:2.0
   BEGIN:VEVENT
   ORGANIZER:Mailto:A@example.com
   ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED;CN=BIG A:Mailto:A@example.com
   ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL;CN=B:Mailto:B@example.com
   ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL;CN=C:Mailto:C@example.com

```
      ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL;CN=Hal:Mailto:D@example.com
      ATTENDEE;RSVP=FALSE;TYPE=ROOM:conf_Big@example.com
      ATTENDEE;ROLE=NON-PARTICIPANT;RSVP=FALSE:Mailto:E@example.com
      DTSTAMP:19970611T190000Z
      DTSTART:19970701T200000Z
      DTEND:19970701T2000000Z
      SUMMARY:Conference
      UID:calsrv.example.com-873970198738777@example.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;PARTSTAT=ACCEPTED:Mailto:B@example.com
      ORGANIZER:MAILTO:A@example.com
      UID:calsrv.example.com-873970198738777@example.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" "partstat" parameter to "declined" or
   "tentative", respectively. Also, "REQUEST-STATUS" is not required in
   successful transactions.

4.2.3 Update An Event

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

```
      BEGIN:VCALENDAR
      PRODID:-//ACME/DesktopCalendar//EN
      METHOD:REQUEST
      VERSION:2.0
      BEGIN:VEVENT
      ORGANIZER:Mailto:A@example.com
```

```
ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:Mailto:A@example.com
ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL:Mailto:B@example.com
ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL:Mailto:C@example.com
ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL;CN=Hal:Mailto:D@example.com
ATTENDEE;ROLE=NON-PARTICIPANT;RSVP=FALSE;
 CUTYPE=ROOM:Mailto:Conf@example.com
ATTENDEE;ROLE=NON-PARTICIPANT;RSVP=FALSE:Mailto:E@example.com
DTSTART:19970701T180000Z
DTEND:19970701T190000Z
SUMMARY:Phone Conference
UID:calsrv.example.com-873970198738777@example.com
SEQUENCE:1
DTSTAMP:19970613T190000Z
STATUS:CONFIRMED
END:VEVENT
END:VCALENDAR
```

4.2.4 Countering an Event Proposal

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

     "A" sends the following "REQUEST":

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:REQUEST
VERSION:2.0
BEGIN:VEVENT
ORGANIZER:Mailto:A@example.com
ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:Mailto:A@example.com
ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL:Mailto:B@example.com
ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL:Mailto:C@example.com
DTSTART:19970701T190000Z
DTEND:19970701T200000Z
SUMMARY:Discuss the Merits of the election results
LOCATION:Green Conference Room
UID:calsrv.example.com-873970198738777a@example.com
SEQUENCE:0
DTSTAMP:19970611T190000Z
STATUS:CONFIRMED
END:VEVENT
END:VCALENDAR
```

     "B" sends "COUNTER" to "A", requesting changes to time and place. "B"
     uses the "COMMENT" property to communicate a rationale for the
     change. Note that the "SEQUENCE" property is NOT incremented on a
     "COUNTER".

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

"A" accepts the changes from "B". To accept a counter-proposal, the
"Organizer" sends a new event "REQUEST" with an incremented sequence
number.

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:REQUEST
VERSION:2.0
BEGIN:VEVENT
ORGANIZER:Mailto:A@example.com
ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:Mailto:A@example.com
ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL:Mailto:B@example.com
ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL:Mailto:C@example.com
DTSTAMP:19970613T190000Z
DTSTART:19970701T160000Z
DTEND:19970701T190000Z
SUMMARY:Discuss the Merits of the election results - changed to
  meet B's schedule
LOCATION:Green Conference Room
UID:calsrv.example.com-873970198738777@example.com
SEQUENCE:1
STATUS:CONFIRMED
END:VEVENT
END:VCALENDAR
```

Instead, "A" rejects "B's" counter proposal

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:DECLINECOUNTER
VERSION:2.0
BEGIN:VEVENT
ORGANIZER:Mailto:A@example.com
ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL:Mailto:B@example.com
COMMENT:Sorry, I cannot change this meeting time
UID:calsrv.example.com-873970198738777@example.com
SEQUENCE:0
DTSTAMP:19970614T190000Z
END:VEVENT
END:VCALENDAR
```

4.2.5 Delegating 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 "Delegate". 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". The "Delegator" MUST:

     . Send a "REPLY" to the "Organizer" with the following updates:
     . The "Delegator's" "ATTENDEE" property "partstat" parameter set
       to "delegated" and the "delegated-to" parameter is set to the
       address of the "Delegate"
     . Add an additional "ATTENDEE" property for the "Delegate" with
       the "delegated-from" property parameter set to the "Delegator"
     . Indicate whether they want to continue to receive updates when
       the "Organizer" sends out updated versions of the event.
       Setting the "rsvp" property parameter to "TRUE" will cause the
       updates to be sent, setting it to "FALSE" causes no further
       updates to be sent. Note that in either case, if the "Delegate"
       declines the invitation the "Delegator" will be notified.
     . The "Delegator" MUST also send a copy of the original "REQUEST"
       method to the "Delegate".

   It is not required that the "Delegate" include the "Delegator" in
   their "REPLY" method. However, it is strongly advised since this will
   inform the "Delegator" whether the "Delegate" plans to attend the
   meeting. [Editors note: How so?] If the "Delegate" declines the
   meeting, the "Delegator" may elect to 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 "C"	
Delegate: "C" delegates to "E"		"C" sends a REPLY to "A" with the ATTENDEE. "partstat" parameter set to "delegated" and with a new "ATTENDEE" property for "E". "E's" ATTENDEE "delegated-from" param is set to "C". "C's" ATTENDEE "delegated-to" param is set to "E". "C" sends REQUEST message to "E" with the original meeting request information. The "partstat" property parameter for "C" is set to "delegated" and the "delegated-to" parameter is set to the address of "E". An "ATTENDEE" property is added for "E" and the "delegated-from" parameter is set to the address of "C".
Confirm meeting attendance		"E" sends REPLY message to "A" and optionally "C" with its "partstat" property parameter set to "ACCEPTED"
Optional: Redistribute meeting to attendees	"A" sends REQUEST message to "B", "C" and "E".	

```
    "C" responds to the "Organizer".

    BEGIN:VCALENDAR
    PRODID:-//ACME/DesktopCalendar//EN
    METHOD:REPLY
    VERSION:2.0
    BEGIN:VEVENT
    ORGANIZER:MAILTO:A@Example.com
    ATTENDEE;PARTSTAT=DELEGATED;DELEGATED-
     TO="Mailto:E@example.com":Mailto:C@example.com
    UID:calsrv.example.com-873970198738777@example.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
    ORGANIZER:Mailto:A@example.com
    ATTENDEE;PARTSTAT=DELEGATED;DELEGATED-
     TO="Mailto:E@example.com":Mailto:C@example.com
    ATTENDEE;RSVP=TRUE;
     DELEGATED-FROM="Mailto:C@example.com":Mailto:E@example.com
    DTSTART:19970701T180000Z
    DTEND:19970701T200000Z
    SUMMARY:Phone Conference
    UID:calsrv.example.com-873970198738777@example.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, "E", sends the following
    message to "A" and "C":

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

```
BEGIN:VEVENT
ORGANIZER:MAILTO:A@Example.com
ATTENDEE;PARTSTAT=ACCEPTED;DELEGATED-
 FROM="Mailto:C@example.com":Mailto:E@example.com
ATTENDEE;PARTSTAT=DELEGATED;
 DELEGATED-TO="Mailto:E@example.com":Mailto:C@example.com
UID:calsrv.example.com-873970198738777@example.com
SEQUENCE:0
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 "partstat" parameter to "DECLINED". The
   "Organizer" SHOULD resend the "REQUEST" to "C" with the "partstat"
   parameter 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 the request or
   delegate it to another CU.

   Response from "E" to "A" and "C". Note the use of the "COMMENT"
   property "E" uses to indicate why the delegation was declined.

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:REPLY
VERSION:2.0
BEGIN:VEVENT
ORGANIZER:MAILTO:A@Example.com
ATTENDEE;PARTSTAT=DELEGATED;
 DELEGATED-TO="Mailto:E@example.com":Mailto:C@example.com
ATTENDEE;PARTSTAT=DECLINED;
 DELEGATED-FROM="Mailto:C@example.com":Mailto:E@example.com
COMMENT:Sorry, I will be out of town at that time.
UID:calsrv.example.com-873970198738777@example.com
SEQUENCE:0
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:REQUEST
      VERSION:2.0
      BEGIN:VEVENT
      ORGANIZER:MAILTO:A@Example.com
      ATTENDEE;PARTSTAT=DECLINED;
       DELEGATED-FROM="Mailto:C@example.com":Mailto:E@example.com
      ATTENDEE;RSVP=TRUE:Mailto:C@example.com
      UID:calsrv.example.com-873970198738777@example.com
      SEQUENCE:0
      SUMMARY:Phone Conference
      DTSTART:19970701T180000Z
      DTEND:19970701T200000Z
      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. The "Organizer" MAY add the forwarded "Calendar User"
   to the attendee list.

4.2.9 Cancel A Group Event

   Individual "A" requests a meeting between individuals "A", "B", "C",
   and "D". 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) MUST be ignored.

```
+------------------------------------------------------------------+
| Action              | "Organizer"        | "Attendee"           |
+------------------------------------------------------------------+
| Initiate a meeting  | "A" sends a REQUEST|                      |
| request             | message to "B", "C",|                     |
|                     | and "D"            |                      |
+------------------------------------------------------------------+
| Decline the meeting |                    | "B" sends a "REPLY"  |
| request             |                    | message to "A" with its |
|                     |                    | "partstat" para-     |
|                     |                    | set to "declined".   |
+------------------------------------------------------------------+
| Cancel the meeting  | "A" sends a CANCEL |                      |
|                     | message to "B", "C" |                     |
|                     | and "D"            |                      |
+------------------------------------------------------------------+
```

The example shows how "A" cancels the event.

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:CANCEL
VERSION:2.0
BEGIN:VEVENT
ORGANIZER:Mailto:A@example.com
ATTENDEE;TYPE=INDIVIDUAL;Mailto:A@example.com
ATTENDEE;TYPE=INDIVIDUAL:Mailto:B@example.com
ATTENDEE;TYPE=INDIVIDUAL:Mailto:C@example.com
ATTENDEE;TYPE=INDIVIDUAL:Mailto:D@example.com
COMMENT:Mr. B cannot attend. It's raining. Lets cancel.
UID:calsrv.example.com-873970198738777@example.com
SEQUENCE:1
STATUS:CANCELLED
DTSTAMP:19970613T190000Z
END:VEVENT
END:VCALENDAR
```

4.2.10 Removing Attendees

   "A" wants to remove "B" from a meeting. This is done by sending a
   "CANCEL" to "B" and removing "B" from the attendee list in the master
   copy of the event.

   +--+
   | Action | "Organizer" | "Attendee" |
   +--+
   | Remove an "B" | "A" sends a CANCEL | |
   | as an "Attendee" | message to "B" | |
   +--+
Update the master	"A" sends the	
copy of the event	updated event to	
	the remaining	
	"Attendees"	
   +--+

   The original meeting includes "A", "B", "C", and "D". The example
   below shows the "CANCEL" that "A" sends to "B". Note that in the
   example below the "STATUS" property is omitted. This is used when the
   meeting itself is cancelled and not when the intent is to remove an
   "Attendee" from the Event.

   BEGIN:VCALENDAR
   PRODID:-//ACME/DesktopCalendar//EN
   METHOD:CANCEL
   VERSION:2.0
   BEGIN:VEVENT
   ORGANIZER:Mailto:A@example.com
   ATTENDEE:mailto:B@example.com
   COMMENT:You're off the hook for this meeting
   UID:calsrv.example.com-873970198738777@example.com
   DTSTAMP:19970613T193000Z
   SEQUENCE:1
   END:VEVENT
   END:VCALENDAR

   The updated master copy of the event is shown below. The "Organizer"
   MAY resend the updated event to the remaining "Attendees". Note that
   "B" has been removed.

   BEGIN:VCALENDAR
   PRODID:-//ACME/DesktopCalendar//EN
   METHOD:REQUEST
   VERSION:2.0
   BEGIN:VEVENT
   ORGANIZER:Mailto:A@example.com

```
     ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:Mailto:A@example.com
     ATTENDEE;TYPE=INDIVIDUAL:Mailto:C@example.com
     ATTENDEE;TYPE=INDIVIDUAL:Mailto:D@example.com
     ATTENDEE;TYPE=ROOM:CR_Big@example.com
     ATTENDEE;ROLE=NON-PARTICIPANT;
      RSVP=FALSE:Mailto:E@example.com
     DTSTAMP:19970611T190000Z
     DTSTART:19970701T200000Z
     DTEND:19970701T203000Z
     SUMMARY:Phone Conference
     UID:calsrv.example.com-873970198738777@example.com
     SEQUENCE:2
     STATUS:CONFIRMED
     END:VEVENT
     END:VCALENDAR
```

4.2.11 Replacing the Organizer

   The scenario for this example begins with "A" as the "Organizer" for
   a recurring meeting with "B", "C", and "D". "A" receives a new job
   offer in another country and drops out of touch. "A" left no
   forwarding address or way to be reached. Using out-of-band
   communication, the other "Attendees" eventually learn what has
   happened and reach an agreement that "B" should become the new
   "Organizer" for the meeting. To do this, "B" sends out a new version
   of the event and the other "Attendees" agree to accept "B" as the new
   "Organizer". "B" also removes "A" from the event.

   When the "Organizer" is replaced, the "SEQUENCE" property value MUST
   be incremented.

   This is the message "B" sends to "C" and "D"

```
     BEGIN:VCALENDAR
     PRODID:-//ACME/DesktopCalendar//EN
     METHOD:REQUEST
     VERSION:2.0
     BEGIN:VEVENT
     ORGANIZER:Mailto:B@example.com
     ATTENDEE;ROLE=CHAIR;STATUS=ACCEPTED:Mailto:B@example.com
     ATTENDEE;TYPE=INDIVIDUAL:Mailto:C@example.com
     ATTENDEE;TYPE=INDIVIDUAL:Mailto:D@example.com
     DTSTAMP:19970611T190000Z
     DTSTART:19970701T200000Z
     DTEND:19970701T203000Z
     RRULE:FREQ=WEEKLY
     SUMMARY:Phone Conference
     UID:123456@example.com
```

```
      SEQUENCE:1
      STATUS:CONFIRMED
      END:VEVENT
      END:VCALENDAR
```

4.3 Busy Time Examples

   Busy time objects can be used in several ways. First, a CU may
   request busy time from another CU for a specific range of time. That
   request can be answered with a busy time Reply. Additionally, a CU
   may simply publish their busy time for a given interval and point
   other CUs to the published location. The following examples outline
   both scenarios.

   Individual "A" publishes busy time for one week.

```
      BEGIN:VCALENDAR
      PRODID:-//ACME/DesktopCalendar//EN
      VERSION:2.0
      METHOD:PUBLISH
      BEGIN:VFREEBUSY
      DTSTAMP:19980101T124100Z
      ORGANIZER:MAILTO:A@Example.com
      DTSTART:19980101T124200Z
      DTEND:19980107T124200Z
      FREEBUSY:19980101T180000Z/19980101T190000Z
      FREEBUSY:19980103T020000Z/19980103T050000Z
      FREEBUSY:19980107T020000Z/19980107T050000Z
      FREEBUSY:19980113T000000Z/19980113T010000Z
      FREEBUSY:19980115T190000Z/19980115T200000Z
      FREEBUSY:19980115T220000Z/19980115T230000Z
      FREEBUSY:19980116T013000Z/19980116T043000Z
      END:VFREEBUSY
      END:VCALENDAR
```

   Individual "A" requests busy time from individuals "B", "C".
   Individual "B" and "C" replies with busy time data to individual "A".
   The following table illustrates the sequence of messages that would
   be exchanged between these individuals.

```
+-----------------------------------------------------------------+
| Action          | "Organizer"        | Attendee                |
+-----------------------------------------------------------------+
| Initiate a busy | "A" sends "REQUEST" |                        |
| time request    | message to "B" and  |                        |
|                 | and "C"             |                        |
+-----------------------------------------------------------------+
| Reply to the "BUSY"|                  | "B" sends a "REPLY"     |
| request with "BUSY"|                  | 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
   ORGANIZER:Mailto:A@example.com
   ATTENDEE;ROLE=CHAIR:Mailto:A@example.com
   ATTENDEE:Mailto:B@example.com
   ATTENDEE:Mailto:C@example.com
   DTSTAMP:19970613T190000Z
   DTSTART:19970701T080000Z
   DTEND:19970701T200000
   UID:calsrv.example.com-873970198738777@example.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
   ORGANIZER:MAILTO:A@example.com
   ATTENDEE:Mailto:B@example.com
   DTSTART:19970701T080000Z
   DTEND:19970701T200000Z
   UID:calsrv.example.com-873970198738777@example.com
   FREEBUSY:19970701T090000Z/PT1H,19970701T140000Z/PT30M

```

```
   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
   TZID:America-SanJose
   TZURL:http://zones.stds_r_us.net/tz/America-SanJose
   BEGIN:STANDARD
   DTSTART:19671029T020000
   RRULE:FREQ=YEARLY;BYDAY=-1SU;BYMONTH=10
   TZOFFSETFROM:-0700
   TZOFFSETTO:-0800
   TZNAME:PST
   END:STANDARD
   BEGIN:DAYLIGHT
   DTSTART:19870405T020000
   RRULE:FREQ=YEARLY;BYDAY=1SU;BYMONTH=4
   TZOFFSETFROM:-0800
   TZOFFSETTO:-0700
   TZNAME:PDT
   END:DAYLIGHT
   END:VTIMEZONE
   BEGIN:VEVENT
   ORGANIZER:Mailto:A@example.com
   ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED;TYPE=INDIVIDUAL:A@example.COM
   ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL:B@example.fr
   ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL:c@example.jp
   DTSTAMP:19970613T190030Z
   DTSTART;TZID=America-SanJose:19970701T140000
   DTEND;TZID=America-SanJose:19970701T150000
   RRULE:FREQ=WEEKLY;INTERVAL=20;WKST=SU;BYDAY=TU
   RDATE;TZID=America-SanJose:19970910T140000
   EXDATE;TZID=America-SanJose:19970909T140000
   EXDATE;TZID=America-SanJose:19971028T140000
   SUMMARY:Weekly Phone Conference
```

```
UID:calsrv.example.com-873970198738777@example.com
SEQUENCE:0
STATUS:CONFIRMED
END:VEVENT
END:VCALENDAR
```

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 PDT.
"Attendee" B@example.fr is in France where the local time on this
date is 9 hours ahead of PDT or 23:00. "Attendee" C@example.jp is in
Japan where local time is 8 hours ahead of UTC or Wednesday, July 2
at 06:00. 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 PDT. The "RDATE" property adds
another instance: WED, 10-SEP-1997 2:00 PM PST.

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

```
TUE, 09-SEP-1997 23:00 GMT
TUE, 09-SEP-1997 14:00 PDT
WED, 10-SEP-1997 06:00 JST
```

and the second is:

```
TUE, 28-OCT-1997 23:00 GMT
TUE, 28-OCT-1997 14:00 PST
WED, 29-OCT-1997 06:00 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
UID:guid-1@host1.com
SEQUENCE:0
RRULE:FREQ=MONTHLY;BYMONTHDAY=1;UNTIL=19980901T210000Z
ORGANIZER:Mailto:A@example.com
ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:Mailto:A@example.com
ATTENDEE:Mailto:B@example.com
ATTENDEE:Mailto:C@example.com
ATTENDEE: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:19970526T083000Z
STATUS:CONFIRMED
END:VEVENT
END:VCALENDAR
```

The event request below is to change the time of a specific instance.
This changes the July 1st instance to July 3rd.

```
BEGIN:VCALENDAR
METHOD:REQUEST
PRODID:-//RDU Software//NONSGML HandCal//EN
VERSION:2.0
BEGIN:VEVENT
UID:guid-1@host1com
RECURRENCE-ID:19970701T210000Z
SEQUENCE:1
ORGANIZER:Mailto:A@example.com
ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:Mailto:A@example.com
ATTENDEE:Mailto:B@example.com
ATTENDEE:Mailto:C@example.com
ATTENDEE: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:19970626T093000Z
STATUS:CONFIRMED
END:VEVENT
END:VCALENDAR
```

4.4.3 Cancel an Instance

   In this example the "Organizer" of a recurring event deletes the
   August 1st instance.

   BEGIN:VCALENDAR
   METHOD:CANCEL
   PRODID:-//RDU Software//NONSGML HandCal//EN
   VERSION:2.0
   BEGIN:VEVENT
   UID:guid-1@host1.com
   ORGANIZER:Mailto:A@example.com
   ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:Mailto:A@example.com
   ATTENDEE:Mailto:B@example.com
   ATTENDEE:Mailto:C@example.com
   ATTENDEE:Mailto:D@example.com
   RECURRENCE-ID:19970801T210000Z
   SEQUENCE:2
   STATUS:CANCELLED
   DTSTAMP:19970721T093000Z
   END:VEVENT
   END:VCALENDAR

4.4.4 Cancel Recurring Event

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

   BEGIN:VCALENDAR
   METHOD:CANCEL
   PRODID:-//RDU Software//NONSGML HandCal//EN
   VERSION:2.0
   BEGIN:VEVENT
   UID:guid-1@host1.com
   ORGANIZER:Mailto:A@example.com
   ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:Mailto:A@example.com
   ATTENDEE:Mailto:B@example.com
   ATTENDEE:Mailto:C@example.com
   ATTENDEE:Mailto:D@example.com
   DTSTAMP:19970721T103000Z
   STATUS:CANCELLED
   SEQUENCE:3
   END:VEVENT
   END:VCALENDAR

4.4.5 Change All Future Instances

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

       BEGIN:VCALENDAR
       METHOD:REQUEST
       PRODID:-//RDU Software//NONSGML HandCal//EN
       VERSION:2.0
       BEGIN:VEVENT
       UID:guid-1@host1.com
       RECURRENCE-ID;THISANDFUTURE:19970901T210000Z
       SEQUENCE:3
       ORGANIZER:Mailto:A@example.com
       ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:Mailto:A@example.com
       ATTENDEE;RSVP=TRUE:Mailto:B@example.com
       ATTENDEE;RSVP=TRUE:Mailto:C@example.com
       ATTENDEE;RSVP=TRUE:Mailto:D@example.com
       DESCRIPTION:IETF-C&S Discussion
       CLASS:PUBLIC
       SUMMARY:IETF Calendaring Working Group Meeting
       DTSTART:19970901T210000Z
       DTEND:19970901T220000Z
       LOCATION:Building 32, Microsoft, Seattle, WA
       DTSTAMP:19970526T083000Z
       STATUS:CONFIRMED
       END:VEVENT
       END:VCALENDAR

4.4.6 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:ADD
       PRODID:-//RDU Software//NONSGML HandCal//EN
       VERSION:2.0
       BEGIN:VEVENT
       UID:123456789@host1.com
       SEQUENCE:4
       ORGANIZER:Mailto:A@example.com
       ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:Mailto:A@example.com
       ATTENDEE;RSVP=TRUE:Mailto:B@example.com
       ATTENDEE;RSVP=TRUE:Mailto:C@example.com
       ATTENDEE;RSVP=TRUE: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:19970629T093000Z
      STATUS:CONFIRMED
      END:VEVENT
      END:VCALENDAR
```

4.4.7 Add A New Series of Instances To A Recurring Event

   The scenario for this example involves an ongoing meeting, originally
   set up to occur every Tuesday. The "Organizer" later decides that
   the meetings need to be on Tuesdays and Thursdays, but does not want
   to reschedule the entire meeting or lose any of the per-instance
   information already collected.

   The original event:

```
      BEGIN:VCALENDAR
      METHOD:REQUEST
      PRODID:-//RDU Software//NONSGML HandCal//EN
      VERSION:2.0
      BEGIN:VEVENT
      UID:123456789@host1.com
      SEQUENCE:0
      RRULE:WKST=SU;BYDAY=TU;FREQ=WEEKLY
      ORGANIZER:Mailto:A@example.com
      ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:Mailto:A@example.com
      ATTENDEE;RSVP=TRUE:Mailto:B@example.com
      SUMMARY:Review Accounts
      DTSTART:19980303T210000Z
      DTEND:19980303T220000Z
      LOCATION:The White Room
      DTSTAMP:19980301T093000Z
      STATUS:CONFIRMED
      END:VEVENT
      END:VCALENDAR
```

   Assume that many other updates happen to this event and that a lot of
   instance-specific information exists in the recurring series. The
   "SEQUENCE" property value is 7 for the next update. Now the
   "Organizer" wants to add Thursdays to the event:

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

```
BEGIN:VEVENT
UID:123456789@host1.com
SEQUENCE:7
RRULE:WKST=SU;BYDAY=TH;FREQ=WEEKLY
ORGANIZER:Mailto:A@example.com
ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:Mailto:A@example.com
ATTENDEE;RSVP=TRUE:Mailto:B@example.com
SUMMARY:Review Accounts
DTSTART:19980303T210000Z
DTEND:19980303T220000Z
DTSTAMP:19980303T193000Z
LOCATION:The Usual conference room
STATUS:CONFIRMED
END:VEVENT
END:VCALENDAR
```

Alternatively, if the "Organizer" is not concerned with per-instance
updates, the entire event can be rescheduled using a "REQUEST". This
is done by using the "UID" of the event to reschedule and including
the modified "RRULE". Note, that since this is an entire rescheduling
of the event, any instance-specific information will be lost.

```
BEGIN:VCALENDAR
METHOD:REQUEST
PRODID:-//RDU Software//NONSGML HandCal//EN
VERSION:2.0
BEGIN:VEVENT
UID:123456789@host1.com
SEQUENCE:7
RRULE:WKST=SU;BYDAY=TU,TH;FREQ=WEEKLY
ORGANIZER:Mailto:A@example.com
ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:Mailto:A@example.com
ATTENDEE;RSVP=TRUE:Mailto:B@example.com
SUMMARY:Review Accounts
DTSTART:19980303T210000Z
DTEND:19980303T220000Z
DTSTAMP:19980303T193000Z
LOCATION:The White Room
STATUS:CONFIRMED
END:VEVENT
END:VCALENDAR
```

The next series of examples illustrate how an "Organizer" would
respond to a "REFRESH" submitted by an "Attendee" after a series of
instance-specific modifications. To convey all instance-specific
changes, the "Organizer" must provide the latest event description
and the relevant instances. The first three examples show the history
including the initial "VEVENT" request and subsequent instance

changes and finally the "REFRESH".

Original Request:

```
BEGIN:VCALENDAR
METHOD:REQUEST
PRODID:-//RDU Software//NONSGML HandCal//EN
VERSION:2.0
BEGIN:VEVENT
UID:123456789@host1.com
SEQUENCE:0
RDATE:19980304T180000Z
RDATE:19980311T180000Z
RDATE:19980318T180000Z
ORGANIZER:Mailto:A@example.com
ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:Mailto:A@example.com
ATTENDEE;RSVP=TRUE:Mailto:B@example.com
SUMMARY:Review Accounts
DTSTART:19980304T180000Z
DTEND:19980304T200000Z
DTSTAMP:19980303T193000Z
LOCATION:Conference Room A
STATUS:CONFIRMED
END:VEVENT
END:VCALENDAR
```

Organizer changes 2nd instance Location and Time:

```
BEGIN:VCALENDAR
METHOD:REQUEST
PRODID:-//RDU Software//NONSGML HandCal//EN
VERSION:2.0
BEGIN:VEVENT
UID:123456789@host1.com
SEQUENCE:1
RECURRENCE-ID:19980311T180000Z
ORGANIZER:Mailto:A@example.com
ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:Mailto:A@example.com
ATTENDEE;RSVP=TRUE:Mailto:B@example.com
SUMMARY:Review Accounts
DTSTART:19980311T160000Z
DTEND:19980311T180000Z
DTSTAMP:19980306T193000Z
LOCATION:The Small conference room
STATUS:CONFIRMED
END:VEVENT
END:VCALENDAR
```

Organizer adds a 4th instance of the meeting using the "ADD" method

```
BEGIN:VCALENDAR
METHOD:ADD
PRODID:-//RDU Software//NONSGML HandCal//EN
VERSION:2.0
BEGIN:VEVENT
UID:123456789@host1.com
SEQUENCE:2
ORGANIZER:Mailto:A@example.com
ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:Mailto:A@example.com
ATTENDEE;RSVP=TRUE:Mailto:B@example.com
SUMMARY:Review Accounts
DTSTART:19980315T180000Z
DTEND:19980315T200000Z
DTSTAMP:19980307T193000Z
LOCATION:Conference Room A
STATUS:CONFIRMED
END:VEVENT
END:VCALENDAR
```

If "B" requests a "REFRESH", "A" responds with the following to
capture all instance-specific data. In this case both the initial
request and an additional "VEVENT" that specifies the instance-
specific data are included. Because these are both of the same type
(they are both "VEVENTS"), they can be conveyed in the same iCalendar
object.

```
BEGIN:VCALENDAR
METHOD:REQUEST
PRODID:-//RDU Software//NONSGML HandCal//EN
VERSION:2.0
BEGIN:VEVENT
UID:123456789@host1.com
SEQUENCE:2
RDATE:19980304T180000Z
RDATE:19980311T160000Z
RDATE:19980315T180000Z
Error! Bookmark not defined.
ORGANIZER:Mailto:A@example.com
ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:Mailto:A@example.com
ATTENDEE;RSVP=TRUE:Mailto:B@example.com
SUMMARY:Review Accounts
DTSTART:19980304T180000Z
DTEND:19980304T200000Z
DTSTAMP:19980303T193000Z
LOCATION:Conference Room A
STATUS:CONFIRMED
```

```
      END:VEVENT

      BEGIN:VEVENT
      Error! Bookmark not defined.
      SEQUENCE:2
      RECURRENCE-ID:19980311T160000Z
      Error! Bookmark not defined.
      ATTENDEE;ROLE=CHAIR;Error! Bookmark not defined.
      ATTENDEE;Error! Bookmark not defined.
      SUMMARY:Review Accounts
      DTSTART:19980311T160000Z
      DTEND:19980304T180000Z
      DTSTAMP:19980306T193000Z
      LOCATION:The Small conference room
      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
      UID:guid-1@host1.com
      RECURRENCE-ID:19970715T210000Z
      SEQUENCE:4
      ORGANIZER:Mailto:A@example.com
      ATTENDEE;ROLE=CHAIR;RSVP=TRUE:Mailto:A@example.com
      ATTENDEE;RSVP=TRUE:Mailto:B@example.com
      ATTENDEE;RSVP=TRUE:Mailto:C@example.com
      ATTENDEE;RSVP=TRUE: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:19970629T094000Z
      END:VEVENT
      END:VCALENDAR
```

4.4.9 Error Reply To A Request

   The following example illustrates a scenario where a meeting is
   proposed containing an unsupported property and a bad property.

   Original Request

   BEGIN:VCALENDAR
   METHOD:REQUEST
   PRODID:-//RDU Software//NONSGML HandCal//EN
   VERSION:2.0
   BEGIN:VEVENT
   UID:guid-1@host1.com
   SEQUENCE:0
   RRULE:FREQ=MONTHLY;BYMONTHDAY=1
   ORGANIZER:Mailto:A@example.com
   ATTENDEE;ROLE=CHAIR:Mailto:A@example.com
   ATTENDEE;RSVP=TRUE:Mailto:B@example.com
   ATTENDEE;RSVP=TRUE:Mailto:C@example.com
   ATTENDEE;RSVP=TRUE:Mailto:D@example.com
   DESCRIPTION:IETF-C&S Conference Call
   CLASS:PUBLIC
   SUMMARY:IETF Calendaring Working Group Meeting
   DTSTART:19970601T210000Z
   DTEND:19970601T220000Z
   DTSTAMP:19970602T094000Z
   LOCATION:Conference Call
   STATUS:CONFIRMED
   FOO:BAR
   END:VEVENT
   END:VCALENDAR

   Response from "B" to indicate that RRULE is not supported and an
   unrecognized property was encountered

   BEGIN:VCALENDAR
   PRODID:-//RDU Software//NONSGML HandCal//EN
   METHOD:REPLY
   VERSION:2.0
   BEGIN:VEVENT
   ORGANIZER:Mailto:A@example.com
   ATTENDEE:Mailto:B@example.com
   REQUEST-STATUS:2.8;Repeating event ignored. Scheduled as a single
     event;RRULE
   REQUEST-STATUS:3.0;Invalid Property Name;FOO
   UID:guid-1@host1.com
   SEQUENCE:0
   DTSTAMP:19970603T094000Z

```
    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 "REQUEST" 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 "partstat" paramater set to "accepted".
Decline the to-do request		"C" sends a REPLY message to "A" with its "partstat" parameter set to "declined".
Tentatively accept the to-do request		"D" sends a REPLY message to "A" with its "partstat" parameter set to "tentative".
Check attendee completion status	"A" sends a REQUEST message to "B" and "D" with current information.	
Attendee indicates percent complete		"B" sends a "REPLY" message indicating percent complete

```
+-------------------------------------------------------------------+
| Attendee indicates  |                      | "D" sends a "REPLY"   |
| completion          |                      | message indicating    |
|                     |                      | completion            |
+-------------------------------------------------------------------+
```

4.5.1 A VTODO Request

   A sample "REQUEST" 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
   ORGANIZER:Mailto:A@example.com
   ATTENDEE;ROLE=CHAIR:Mailto:A@example.com
   ATTENDEE;RSVP=TRUE:Mailto:B@example.com
   ATTENDEE;RSVP=TRUE:Mailto:C@example.com
   ATTENDEE;RSVP=TRUE:Mailto:D@example.com
   DTSTART:19970701T170000Z
   DUE:19970722T170000Z
   PRIORITY:1
   SUMMARY:Create the requirements document
   UID:calsrv.example.com-873970198738777-00@example.com
   SEQUENCE:0
   DTSTAMP:19970717T200000Z
   STATUS:Needs Action
   END:VTODO
   END:VCALENDAR

4.5.2 A VTODO Reply

   "B" accepts the to-do.

   BEGIN:VCALENDAR
   PRODID:-//ACME/DesktopCalendar//EN
   METHOD:REPLY
   VERSION:2.0
   BEGIN:VTODO
   ORGANIZER:Mailto:A@example.com
   ATTENDEE;PARTSTAT=ACCEPTED:Mailto:B@example.com
   UID:calsrv.example.com-873970198738777-00@example.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 TODO or indicated tentative acceptance by
   setting the "partstat" property parameter sequence to "declined" or
   "tentative", respectively.

4.5.3 A VTODO Request for Updated Status

   "A" requests status from all "Attendees".

```
   BEGIN:VCALENDAR
   PRODID:-//ACME/DesktopCalendar//EN
   METHOD:REQUEST
   VERSION:2.0
   BEGIN:VTODO
   ORGANIZER:Mailto:A@example.com
   ATTENDEE;ROLE=CHAIR:Mailto:A@example.com
   ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL:Mailto:B@example.com
   ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL:Mailto:D@example.com
   UID:calsrv.example.com-873970198738777-00@example.com
   SUMMARY:Create the requirements document
   PRIORITY:1
   SEQUENCE:0
   STATUS:IN-PROCESS
   DTSTART:19970701T170000Z
   DTSTAMP:19970717T230000Z
   END:VTODO
   END:VCALENDAR
```

4.5.4 A Reply: Percent-Complete

   A reply indicating the task 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
   ORGANIZER:MAILTO:A@example.com
   ATTENDEE;PARTSTAT=IN-PROCESS:Mailto:B@example.com
   PERCENT-COMPLETE:75
   UID:calsrv.example.com-873970198738777-00@example.com
   DTSTAMP:19970717T233000Z
   SEQUENCE:0
   END:VTODO
   END:VCALENDAR
```

4.5.5 A Reply: Completed

   A reply indicating that "D" completed "D's" part of the assignment.

   BEGIN:VCALENDAR
   PRODID:-//ACME/DesktopCalendar//EN
   METHOD:REPLY
   VERSION:2.0
   BEGIN:VTODO
   ORGANIZER:MAILTO:A@example.com
   ATTENDEE;PARTSTAT=COMPLETED:Mailto:D@example.com
   UID:calsrv.example.com-873970198738777-00@example.com
   DTSTAMP:19970717T233000Z
   SEQUENCE:0
   END:VTODO
   END:VCALENDAR

4.5.6 An Updated VTODO Request

   Organizer "A" resends the "VTODO" calendar component. "A" sets the
   overall completion for the to-do at 40%.

   BEGIN:VCALENDAR
   PRODID:-//ACME/DesktopCalendar//EN
   METHOD:REQUEST
   VERSION:2.0
   BEGIN:VTODO
   ORGANIZER:Mailto:A@example.com
   ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:Mailto:A@example.com
   ATTENDEE;PARTSTAT=ACCEPTED;TYPE=INDIVIDUAL:Mailto:B@example.com
   ATTENDEE;PARTSTAT=IN-PROCESS;TYPE=INDIVIDUAL:Mailto:D@example.com
   DTSTART:19970701T170000Z
   DUE:19970722T170000Z
   PRIORITY:1
   SUMMARY:Create the requirements document
   UID:calsrv.example.com-873970198738777-00@example.com
   SEQUENCE:1
   DTSTAMP:19970718T100000Z
   STATUS:IN-PROGRESS
   PERCENT-COMPLETE:40
   END:VTODO
   END:VCALENDAR

4.5.7 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 "D".

   BEGIN:VCALENDAR
   PRODID:-//ACME/DesktopCalendar//EN
   METHOD:REQUEST
   VERSION:2.0
   BEGIN:VTODO
   ORGANIZER:Mailto:A@example.com
   ATTENDEE;ROLE=CHAIR:Mailto:A@example.com
   ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL:Mailto:B@example.com
   ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL:Mailto:D@example.com
   RRULE:FREQ=MONTHLY;COUNT=10;BYDAY=1FR
   DTSTART:19980101T100000-0700
   DUE:19980103T100000-0700
   SUMMARY:Send Status Reports to Area Managers
   UID:calsrv.example.com-873970198738777-00@example.com
   SEQUENCE:0
   DTSTAMP:19970717T200000Z
   STATUS:NEEDS ACTION
   PRIORITY:1
   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 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 is applied to each
   recurring instance of the "VTODO" calendar component to determine the
   "DUE" date of the instance.

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;PARTSTAT=IN-PROCESS:Mailto:B@example.com
   PERCENT-COMPLETE:75
   UID:calsrv.example.com-873970198738777-00@example.com
   DTSTAMP:19970717T233000Z
   RECURRENCE-ID:19980101T170000Z
   SEQUENCE:1
   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
   METHOD:PUBLISH
   PRODID:-//ACME/DesktopCalendar//EN
   VERSION:2.0
   BEGIN:VJOURNAL
   DTSTART:19971002T200000Z
   ORGANIZER:MAILTO:A@Example.com
   SUMMARY:Phone conference minutes
   DESCRIPTION:The editors meeting was held on October 1, 1997.
     Details are in the attached document.
   UID:0981234-1234234-2410@example.com
   RELATED-TO:0981234-1234234-2402-35@example.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
   ORGANIZER:Mailto:A@example.com
   ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:Mailto:A@example.com
   ATTENDEE:Mailto:B@example.com
   ATTENDEE:Mailto:C@example.com
```

```
      ATTENDEE:Mailto:D@example.com
      UID: guid-1-12345@host1.com
      DTSTAMP:19970603T094000
      END:VEVENT
      END:VCALENDAR
```

4.7.2 Bad RECURRENCE-ID

   Component instances are identified by the combination of "UID",
   "RECURRENCE-ID", and "SEQUENCE". When an "Organizer" sends a request
   to an "Attendee", there are three cases in which an instance cannot
   be found. They are:

   1.   The component with the referenced "UID" and "RECURRENCE-ID" has
        been found but the "SEQUENCE" number in the calendar store does
        not match that of the ITIP message.

   2.   The component with the referenced "UID" has been found, the
        "SEQUENCE" numbers match, but the "RECURRENCE-ID" cannot be
        found.

   3.   The "UID" and "SEQUENCE" numbers are found but the CUA does not
        support recurrences.

   In case (1), two things can happen. If the "SEQUENCE" number of the
   "Attendee's" instance is larger than that in the "Organizer's"
   message then the "Attendee" is receiving an out-of-sequence message
   and MUST ignore it. If the "SEQUENCE" number of the "Attendee's"
   instance is smaller, then the "Organizer" is sending out a newer
   version of the component and the "Attendee's" version needs to be
   updated. Since one or more updates have been missed, the "Attendee"
   SHOULD send a "REFRESH" message to the "Organizer" to get an updated
   version of the event.

   In case (2), something has gone wrong. Both the "Organizer" and the
   "Attendee" should have the same instances, but the "Attendee" does
   not have the referenced instance. In this case the "Attendee" SHOULD
   send a "REFRESH" to the "Organizer" to get an updated version of the
   event.

   In case (3), the limitations of the "Attendee's" CUA makes it
   impossible to match an instance other than the single instance
   scheduled. In this case, the "Attendee" need not send a "REFRESH" to
   the "Organizer".

   The example below shows a sequence in which an "Attendee" sends a
   "REFRESH" to the "Organizer".

```
+---------------------------------------------------------------+
| Action              | "Organizer"        | Attendee          |
+---------------------------------------------------------------+
| Update an instance  | "A" sends  "REQUEST"|                  |
| request             | message to "B"     |                   |
+---------------------------------------------------------------+
| Attendee requests   |                    | "B" sends a "REFRESH" |
| refresh because     |                    | message to "A"    |
| "RECURRENCE-ID" was |                    |                   |
| not found           |                    |                   |
+---------------------------------------------------------------+
| Refresh the entire  | "A" sends the      |                   |
| Event               | latest copy of the |                   |
|                     | Event to "B"       |                   |
+---------------------------------------------------------------+
| Attendee handles    |                    | "B" updates to the |
| the request and     |                    | latest copy of the |
| updates the         |                    | meeting.          |
| instance            |                    |                   |
+---------------------------------------------------------------+
```

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
ORGANIZER:Mailto:A@example.com
ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED:Mailto:A@example.com
ATTENDEE:Mailto:B@example.com
DESCRIPTION:IETF-C&S Conference Call
SUMMARY:IETF Calendaring Working Group Meeting
DTSTART:19970801T210000Z
DTEND:19970801T220000Z
RECURRENCE-ID:19970809T210000Z
DTSTAMP:19970726T083000
STATUS:CONFIRMED
END:VEVENT
END:VCALENDAR
```

"B" has the event with "UID" property "acme-12345@host1.com" but
"B's" "SEQUENCE" property value is "1" and the event does not have an
instance at the specified recurrence time. This means that "B" has

missed at least one update and needs a new copy of the event. "B"
requests the latest copy of the event with the following refresh
message:

```
BEGIN:VCALENDAR
PRODID:-//RDU Software//NONSGML HandCal//EN
METHOD:REFRESH
VERSION:2.0
BEGIN:VEVENT
ORGANIZER:Mailto:A@example.com
ATTENDEE: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
REQUEST           PUBLISH
REPLY             Required
ADD               Required
CANCEL            Required
REFRESH           Required
COUNTER           Reply with Not Supported
DECLINECOUNTER    Required if EVENT-COUNTER is implemented; otherwise
                  reply with Not Supported

iCalendar
Property          Fallback
-------------     --------------------------------------------------------
CALSCALE          Ignore; assume GREGORIAN
PRODID            Ignore
METHOD            Required as described in the Method list above
VERSION           Ignore
```

```
Event-Related
Components         Fallback
--------------    ------------------------------------------------------
VALARM            Reply with Not Supported
VTIMEZONE         Required if any DateTime value refers to a time zone.


Component
Property          Fallback
--------------    ------------------------------------------------------
ATTACH            Ignore
ATTENDEE          Required if EVENT-REQUEST is not implemented;
                  otherwise reply with Not Supported
CATEGORIES        Ignore
CLASS             Ignore
COMMENT           Ignore
COMPLETED         Ignore
nCONTACT           Ignore
CREATED           Ignore
DESCRIPTION       Required
DURATION          Reply with Not Supported
DTSTAMP           Required
DTSTART           Required
DTEND             Required
EXDATE            Ignore
EXRULE            Ignore Reply with Not Supported. If implemented,
                  VTIMEZONE MUST also be implemented.
GEO               Ignore
LAST-MODIFIED     Ignore
LOCATION          Required
ORGANIZER         Ignore
PRIORITY          Ignore
RELATED-TO        Ignore
RDATE             Ignore
RRULE             Ignore. The first instance occurs on the DTStart
                  property. If implemented, VTIMEZONE MUST also be
                  implemented.
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
URL               Ignore
UID               Required
X-                Ignore
```

5.1.2 Free/Busy-Related Fallbacks

Method	Fallback
PUBLISH	Implementations MAY ignore the METHOD type. The REQUEST-STATUS "3.14;Unsupported capability" MUST be returned.
REQUEST	Implementations MAY ignore the METHOD type. The REQUEST-STATUS "3.14;Unsupported capability" MUST be returned.
REPLY	Implementations MAY ignore the METHOD type. The REQUEST-STATUS "3.14;Unsupported capability" MUST be returned.

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

Component Property	Fallback
COMMENT	Ignore
CONTACT	Ignore
DTEND	Required
DTSTAMP	Required
DTSTART	Required
DURATION	Required
FREEBUSY	Required
ORGANIZER	Ignore
REQUEST-STATUS	Ignore
UID	Required
URL	Ignore
X-	Ignore

5.1.3 To-Do-Related Fallbacks

Method	Fallback
PUBLISH	Required
REQUEST	PUBLISH
REPLY	Required
ADD	Required

```
CANCEL             Required
REFRESH            Required
COUNTER            Reply with Not Supported
DECLINECOUNTER     Required if VTODO - COUNTER is implemented; otherwise
                   reply with Not Supported


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


To-Do-Related
Components         Fallback
--------------     --------------------------------------------------------
VALARM             Reply with Not Supported
VTIMEZONE          Required if any DateTime value refers to a time zone.


Component
Property           Fallback
--------------     --------------------------------------------------------
ATTACH             Ignore
ATTENDEE           Required if REQUEST is not implemented; otherwise
                   ignore
CATEGORIES         Ignore
CLASS              Ignore
COMMENT            Ignore
COMPLETED          Required
CONTACT            Ignore
CREATED            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. If implemented,
                   VTIMEZONE MUST also be implemented.
LAST-MODIFIED      Ignore
LOCATION           Ignore
ORGANIZER          Ignore
PERCENT-COMPLETE   Ignore
PRIORITY           Required
RECURRENCE-ID      Ignore
```

```
RELATED-TO         Ignore
REQUEST-STATUS     Ignore
RDATE              Ignore
RRULE              Ignore.  The first instance occurs on the DTSTART
                   property. If implemented, VTIMEZONE MUST also be
                   implemented.
RESOURCES          Ignore
SEQUENCE           Required
STATUS             Required
SUMMARY            Ignore
URL                Ignore
UID                Required
X-                 Ignore
```

5.1.4 Journal-Related Fallbacks

Method	Fallback
PUBLISH	Implementations MAY ignore the METHOD type. The REQUEST-STATUS "3.14;Unsupported capability" MUST be returned.
ADD	Implementations MAY ignore the METHOD type. The REQUEST-STATUS "3.14;Unsupported capability" MUST be returned.
CANCEL	Implementations MAY ignore the METHOD type. The REQUEST-STATUS "3.14;Unsupported capability" MUST be returned.

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

Journal-Related Components	Fallback
VTIMEZONE	Required if any DateTime value refers to a time zone.

```
Component
Property        Fallback
--------------  --------------------------------------------------
ATTACH          Ignore
ATTENDEE        Required if JOURNAL-REQUEST is implemented; otherwise
                ignore
CATEGORIES      Ignore
CLASS           Ignore
COMMENT         Ignore
CONTACT         Ignore
CREATED         Ignore
DESCRIPTION     Required
DTSTAMP         Required
DTSTART         Required
EXDATE          Ignore
EXRULE          Ignore Reply with Not Supported. If implemented,
                VTIMEZONE MUST also be implemented.
LAST-MODIFIED   Ignore
ORGANIZER       Ignore
RECURRENCE-ID   Ignore
RELATED-TO      Ignore
RDATE           Ignore.
RRULE           Ignore. The first instance occurs on the DTSTART
                property. If implemented, VTIMEZONE MUST also be
                implemented.
SEQUENCE        Required
STATUS          Ignore
SUMMARY         Required
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, a "CANCEL" method may be received
   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 CU they MUST send a
   "REPLY" method to the "Organizer" using the "ATTENDEE" properties to
   indicate that the request was delegated and to whom. Hence, it is
   possible for an "Organizer" to receive an "REPLY" from a CU 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 "REFRESH" message 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.

6 Security Considerations

   ITIP is 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 iTIP, the "Organizer" (or someone working on the "Organizer's"
   behalf) is the only person authorized to make changes to an existing
   "VEVENT", "VTODO", "VJOURNAL" calendar component and republish it or
   redistribute 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 republished or sent to the "Attendees".

6.1.2 Spoofing the "Attendee"

   In iTIP, an "Attendee" of a "VEVENT" or "VTODO" calendar component
   (or someone working on the "Attendee's" behalf) 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 Unauthorized Replacement of the Organizer

   There will be circumstances when "Attendees" of an event or to-do
   decide, using out-of-band mechanisms, that the "Organizer" must be
   replaced. When the new "Organizer" sends out the updated "VEVENT" or
   "VTODO" the "Attendee's" CUA will detect that the "Organizer" has
   been changed, but it has no way of knowing whether or not the change
   was mutually agreed upon.

6.1.4 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.

6.1.5 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.

6.1.6   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 an 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.

6.1.7 Unauthorized REFRESH Requests

   It is possible for an "Organizer" to receive a "REFRESH" request from
   someone who is not an "Attendee" of an event or to-do. Only
   "Attendee's" of an event or to-do are authorized to receive replies
   to "REFRESH" requests. Replying to such requests to anyone who is not
   an "Attendee" may be a security problem.

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 encrypted. This may be accomplished using
   public key technology, specifically Security Multiparts for MIME

[RFC-1847] in the iTIP transport binding. This helps mitigate the
threats of spoofing, eavesdropping and malicious changes in transit.

6.2.1 Use of [RFC-1847] to secure iTIP transactions

iTIP transport bindings MUST provide a mechanism based on Security
Multiparts for MIME [RFC-1847] 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.

Implementations MAY provide controls for users to disable this
capability.

6.2.2 Implementation Controls

The threat of unauthorized replacement of the "Organizer" SHOULD be
mitigated by a calendar system that uses this protocol by providing
controls or alerts that make "Calendar Users" aware of such
"Organizer" changes and allowing them to decide whether or not the
request should be honored.

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.

The threat of unauthorized "REFRESH" requests SHOULD be mitigated by
a calendar system that uses this protocol by providing controls or
alerts that allow "Calendar User" to decide whether or not the
request should be honored. An implementation MAY decide to maintain,
for audit or historical purposes, "Calendar Users" who were part of
an attendee list and who were subsequently uninvited. Similar
controls or alerts should be provided when a "REFRESH" request is
received from these "Calendar Users" as well.

7 Acknowledgments

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

Anik Ganguly, Dan Hickman, Paul Hill, Daryl Huff, Bruce Kahn, Antoine
Leca, Bob Mahoney, John Noerenberg, Leo Parker, John Rose, Doug
Royer, Vinod Seraphin, Richard Shusterman, Derik Stenerson, John Sun,
Alexander Taler, Kevin Tsurutome.

8 Bibliography

    [iCAL]   Dawson, F. and D. Stenerson, "Internet Calendaring and
                Scheduling Core Object Specification - iCalendar", RFC
                2445, November 1998.

    [iMIP]   Dawson, F., Mansour, S. and S. Silverberg, "iCalendar
                Message-Based Interoperability Protocol - iMIP", RFC 2447,
                November 1998.

    [RFC-2119] Bradner, S., "Key words for use in RFCs to Indicate
                Requirement Levels", BCP 14, RFC 2119, March 1997.

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

9 Authors' Addresses

   The following address information is provided in a vCard v3.0,
   Electronic Business Card, format.

   The authors of this memo are:

   BEGIN:VCARD
   VERSION:3.0
   N:Dawson;Frank
   FN:Frank Dawson
   ORG:Lotus Development Corporation
   ADR;WORK;POSTAL;PARCEL:;;6544 Battleford Drive;Raleigh;NC;27613-
    3502;USA
   TEL;TYPE=WORK,MSG:+1-919-676-9515
   TEL;TYPE=WORK,FAX:+1-919-676-9564
   EMAIL;TYPE=PREF,INTERNET:Frank_Dawson@Lotus.com
   EMAIL;TYPE=INTERNET:fdawson@earthlink.net
   URL:http://home.earthlink.net/~fdawson
   END:VCARD

   BEGIN:VCARD
   VERSION:3.0
   N:Hopson;Ross
   FN:Ross Hopson
   ORG:On Technology, Inc.
   ADR;TYPE=WORK,POSTAL,PARCEL:;Suite 1600;434 Fayetteville St.
     Mall\, Two Hannover Square;Raleigh;NC;27601
   TEL;TYPE=WORK,MSG:+1-919-890-4036
   TEL;TYPE=WORK,FAX:+1-919-890-4100
   EMAIL;TYPE=INTERNET:rhopson@on.com
   END:VCARD

   BEGIN:VCARD
   VERSION:3.0
   N:Mansour;Steve
   FN:Steve Mansour
   ORG:Netscape Communications Corporation
   ADR;TYPE=WORK,POSTAL,PARCEL:;;501 East Middlefield Road;Mountain
     View;CA;94043;USA
   TEL;TYPE=WORK,MSG:+1-650-937-2378
   TEL;TYPE=WORK,FAX:+1-650-937-2103
   EMAIL;TYPE=INTERNET:sman@netscape.com
   END:VCARD

```
BEGIN:VCARD
VERSION:3.0
N:Silverberg;Steve
FN:Steve Silverberg
ORG:Microsoft Corporation
ADR;TYPE=WORK,POSTAL,PARCEL:;;One Microsoft Way;
Redmond;WA;98052-6399;USA
TEL;TYPE=WORK,MSG:+1-425-936-9277
TEL;TYPE=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
VERSION:3.0
N:Ganguly;Anik
FN:Anik Ganguly
ORG:Open Text Inc.
ADR;TYPE=WORK,POSTAL,PARCEL:;Suite 101;38777 West Six Mile Road;
 Livonia;MI;48152;USA
TEL;TYPE=WORK,MSG:+1-734-542-5955
EMAIL;TYPE=INTERNET:ganguly@acm.org
END:VCARD
```

The co-chairman of that working group is:

```
BEGIN:VCARD
VERSION:3.0
N:Moskowitz;Robert
FN:Robert Moskowitz
NICKNAME:Bob
EMAIL; TYPE=INTERNET:rgm-ietf@htt-consult.com
END:VCARD
```

10. Full Copyright Statement