Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: November 19, 2011

C. Daboo
Apple, Inc.
M. Douglass
RPI
S. Lees
Microsoft

May 18, 2011

xCal: The XML format for iCalendar
draft-daboo-et-al-icalendar-in-xml-09

Abstract

This specification defines "xCal", an XML format for iCalendar data.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on November 19, 2011.

Table of Contents

1. Introduction

The iCalendar data format [RFC5545] is a widely deployed interchange format for calendaring and scheduling data. While many applications and services consume and generate calendar data, iCalendar is a specialized format that requires its own parser/generator. In contrast, XML-based formats are widely used for interoperability between applications, and the many tools that generate, parse, and manipulate XML make it easier to work with than iCalendar.

The purpose of this specification is to define "xCal", an XML format for iCalendar data. xCal is defined as a straightforward mapping into XML from iCalendar, so that iCalendar data can be converted to XML, and then back to iCalendar, without losing any semantic meaning in the data. Anyone creating xCal calendar data according to this specification will know that their data can be converted to a valid iCalendar representation as well.

Key design considerations are:

Round-tripping (converting an iCalendar instance to xCal and back) will give the same result as the starting point.

Preserve the semantics of the iCalendar data. While a simple consumer can easily browse the calendar data in xCal, a full understanding of iCalendar is still required in order to modify and/or fully comprehend the calendar data.

Ability to handle many extensions to the underlying iCalendar specification without requiring an update to this document.

2. Conventions Used in This Document

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

When XML element types in the namespace "urn:ietf:params:xml:ns:icalendar-2.0" are referenced in this document outside of the context of an XML fragment, the string "IC:" will be prefixed to the element types.

Some examples in this document contain "partial" XML documents used for illustrative purposes. In these examples, three periods "..." are used to indicate a portion of the document that has been removed for compactness.

3. Converting from iCalendar to xCal

This section describes how iCalendar data is converted to xCal using a simple mapping between the iCalendar data model and XML elements.

3.1. Pre-processing

iCalendar uses a line folding mechanism to limit lines of data to a maximum line length (typically 72 characters) to ensure maximum likelihood of preserving data integrity as it is transported via various means (e.g., email) - see Section 3.1 of [RFC5545]. Prior to converting iCalendar data into xCal all folded lines MUST be unfolded.

iCalendar data uses an "escape" character sequence for text values and property parameter values. When such text elements are converted into xCal the escaping MUST be removed.

iCalendar uses a base64 encoding for binary data. However, it does not restrict the encoding from being applied to non-binary value types. So the following rules MUST be applied when processing a property with the "ENCODING" property parameter set to "BASE64":

- If the property value type is "BINARY", the base64 encoding MUST be preserved.

- If the value type is not "BINARY", the "ENCODING" property parameter MUST be removed, and the value MUST be base64 decoded.

One key difference in the formatting of values used in iCalendar and xCal is that in xCal the specification use date/time and utc-offset values aligned with the syntax of [W3C.REC-xmlschema-2-20041028] to aid with XML processing.

3.2. iCalendar stream (RFC5545 section 3.4)

At the top level of the iCalendar object model is an "iCalendar stream". This object encompasses multiple "iCalendar objects". In xCal, the entire stream is contained in the root IC:icalendar XML element.

An iCalendar stream can contain one or more iCalendar objects. Each iCalendar object, delimited by "BEGIN:VCALENDAR" and "END:VCALENDAR", is enclosed by the IC:vcalendar XML element.

Example:

```
<?xml version="1.0" encoding="utf-8"?>
<icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
  <vcalendar>
  ...
  </vcalendar>
</icalendar>
```

iCalendar objects are comprised of a set of "components", "properties", "parameters" and "values". A "component" can contain other "components" or "properties". A "property" has a value and optionally a set of "parameters".

In xCal, component elements, for example IC:vevent, IC:vtodo, are contained within an IC:components XML element. Within the component element, another IC:components element could appear (representing components nested within components) or the IC:properties XML element could appear. IC:properties is used to encapsulate iCalendar properties.

Each iCalendar property will be mapped to its own XML element as described below. Within each of these elements there is an optional IC:parameters XML element used to encapsulate any iCalendar property parameters. Additionally there will be one or more XML elements representing the value of the iCalendar property.

Example:

```
<?xml version="1.0" encoding="utf-8"?>
<icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
  <vcalendar>
    <properties>
    ...
    </properties>
    <components>
    ...
    </components>
  </vcalendar>
</icalendar>
```

Item	XML element	XML Definition
iCalendar Stream	IC:icalendar	Appendix A # 3.4
VCALENDAR	IC:vcalendar	Appendix A # 3.6

3.3. Components (RFC5545 section 3.6)

Each calendar component in the "VCALENDAR" object, delimited by "BEGIN" and "END", will be converted to an enclosing XML element with the same name, but in lowercase. Any iCalendar components added in the future will be converted in the same way. As a non-normative reference, the table below shows the current iCalendar components and their xCal equivalents.

Component	XML element	XML Definition
VEVENT	IC:vevent	Appendix A # 3.6.1
VTODO	IC:vtodo	Appendix A # 3.6.2
VJOURNAL	IC:vjournal	Appendix A # 3.6.3
VFREEBUSY	IC:vfreebusy	Appendix A # 3.6.4
VTIMEZONE	IC:vtimezone	Appendix A # 3.6.5
STANDARD	IC:standard	Appendix A # 3.6.5
DAYLIGHT	IC:daylight	Appendix A # 3.6.5
VALARM	IC:valarm	Appendix A # 3.6.6

3.4. Properties (RFC5545 section 3.7 and 3.8)

iCalendar properties , whether they apply to the "VCALENDAR" object or to a component, are handled in a consistent way in the xCal format.

iCalendar properties are enclosed in the XML element IC:properties.

Each individual iCalendar property is represented in xCal by an element of the same name as the iCalendar property, but in lowercase. For example, the "CALSCALE" property is represented in xCal by the IC:calscale element.

Example:

```
<?xml version="1.0" encoding="utf-8"?>
<icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
  <vcalendar>
    <properties>
      <calscale>...</calscale>
      <version>...</version>
      <prodid>...</prodid>
    </properties>
    <components>
    ...
    </components>
  </vcalendar>
</icalendar>
```

Each property can contain an IC:parameters XML element encapsulating any iCalendar property parameters associated with the iCalendar property.

Each property will contain one or more "value" XML elements as described below representing the value of the iCalendar property. As with components, the table is a non-normative reference and is not meant to be exclusive. Any iCalendar properties added in the future will be converted using the same mapping rule.

Property	XML element	XML Definition
CALSCALE	IC:calscale	Appendix A # 3.7.1
METHOD	IC:method	Appendix A # 3.7.2
PRODID	IC:prodid	Appendix A # 3.7.3
VERSION	IC:version	Appendix A # 3.7.4
ATTACH	IC:attach	Appendix A # 3.8.1.1
CATEGORIES	IC:categories	Appendix A # 3.8.1.2
CLASS	IC:class	Appendix A # 3.8.1.3
COMMENT	IC:comment	Appendix A # 3.8.1.4
DESCRIPTION	IC:description	Appendix A # 3.8.1.5
GEO	IC:geo	Appendix A # 3.8.1.6
LOCATION	IC:location	Appendix A # 3.8.1.7
PERCENT-COMPLETE	IC:percent-complete	Appendix A # 3.8.1.8
PRIORITY	IC:priority	Appendix A # 3.8.1.9
RESOURCES	IC:resources	Appendix A # 3.8.1.10
STATUS	IC:status	Appendix A # 3.8.1.11
SUMMARY	IC:summary	Appendix A # 3.8.1.12
COMPLETED	IC:completed	Appendix A # 3.8.2.1
DTEND	IC:dtend	Appendix A # 3.8.2.2
DUE	IC:due	Appendix A # 3.8.2.3
DTSTART	IC:dtstart	Appendix A # 3.8.2.4
DURATION	IC:duration	Appendix A # 3.8.2.5

FREEBUSY	IC:freebusy	Appendix A # 3.8.2.6
TRANSP	IC:transp	Appendix A # 3.8.2.7
TZID	IC:tzid	Appendix A # 3.8.3.1
TZNAME	IC:tzname	Appendix A # 3.8.3.2
TZOFFSETFROM	IC:tzoffsetfrom	Appendix A # 3.8.3.3
TZOFFSETTO	IC:tzoffsetto	Appendix A # 3.8.3.4
TZURL	IC:tzurl	Appendix A # 3.8.3.5
ATTENDEE	IC:attendee	Appendix A # 3.8.4.1
CONTACT	IC:contact	Appendix A # 3.8.4.2
ORGANIZER	IC:organizer	Appendix A # 3.8.4.3
RECURRENCE-ID	IC:recurrence-id	Appendix A # 3.8.4.4
RELATED-TO	IC:related-to	Appendix A # 3.8.4.5
URL	IC:url	Appendix A # 3.8.4.6
UID	IC:uid	Appendix A # 3.8.4.7
EXDATE	IC:exdate	Appendix A # 3.8.5.1
RDATE	IC:rdate	Appendix A # 3.8.5.2
RRULE	IC:rrule	Appendix A # 3.8.5.3
ACTION	IC:action	Appendix A # 3.8.6.1
REPEAT	IC:repeat	Appendix A # 3.8.6.2
TRIGGER	IC:trigger	Appendix A # 3.8.6.3
CREATED	IC:created	Appendix A # 3.8.7.1
DTSTAMP	IC:dtstamp	Appendix A # 3.8.7.2
LAST-MODIFIED	IC:last-modified	Appendix A # 3.8.7.3
SEQUENCE	IC:sequence	Appendix A # 3.8.7.4
REQUEST-STATUS	IC:request-status	Appendix A # 3.8.8.3

3.4.1. Special Cases for Properties

This section describes some properties that have special handling when converting to xCal.

3.4.1.1. Multi-valued Properties

The following iCalendar properties can have values that consist of a list of "standard" iCalendar values separated by a specific delimiter. In xCal these properties are represented by an XML element that contains multiple "value" elements (Section 3.6).

Property	XML element	XML Definition
CATEGORIES	IC:categories	Appendix A # 3.8.1.2
RESOURCES	IC:resources	Appendix A # 3.8.1.10
FREEBUSY	IC:freebusy	Appendix A # 3.8.2.6
EXDATE	IC:exdate	Appendix A # 3.8.5.1
RDATE	IC:rdate	Appendix A # 3.8.5.2

3.4.1.2. GEO Property

In iCalendar, the "GEO" property value is defined as a semi-colon separated list of two "FLOAT" values, the first representing latitude and the second longitude.

In xCal, the value for the IC:geo element is represented by two XML elements. These are an IC:latitude element and an IC:longitude element, each of which contain float values. See Appendix A # 3.8.1.6.

Example:

```
<?xml version="1.0" encoding="utf-8"?>
<icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
  ...
  <geo>
    <latitude>37.386013</latitude>
    <longitude>-122.082932</longitude>
  </geo>
  ...
</icalendar>
```

3.4.1.3. REQUEST-STATUS Property

In iCalendar, the "REQUEST-STATUS" property value is defined as a semi-colon separated list of two or three "TEXT" values. The first represents a code, the second a description, and the third (optional) additional data.

In xCal, the value for the IC:request-status element is represented by two or three XML elements. These are an IC:code element, an IC: description element, and optionally an IC:data element, each of which contain the corresponding "TEXT" values. See Appendix A # 3.8.8.3.

Example:

```
<?xml version="1.0" encoding="utf-8"?>
<icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
  ...
  <request-status>
    <code>2.0</code>
    <description>Success</description>
  </request-status>
  ...
</icalendar>
```

3.5. Parameters (RFC5545 section 3.2)

iCalendar property parameters are enclosed in the XML element IC:parameters which optionally occurs once in each property XML element.

Each individual iCalendar property parameter is represented in xCal by an element of the same name as the iCalendar property parameter, but in lowercase. For example, the "PARTSTAT" property parameter is represented in xCal by the IC:partstat element.

Example:

```
<?xml version="1.0" encoding="utf-8"?>
<icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
  <vcalendar>
    ...
    <components>
      ...
      <attendee>
        <parameters>
          <partstat><text>NEEDS-ACTION</text></partstat>
        </parameters>
        ...
      </attendee>
      ...
    </components>
  </vcalendar>
</icalendar>
```

Each XML parameter element contains one or more child XML elements representing iCalendar value types.

As with components and properties, the table below is shown as a non-normative reference. Any property parameters added to iCalendar in the future will be converted by the rule given above.

Parameter	XML element	XML Definition
ALTREP	IC:altrep	Appendix A # 3.2.1
CN	IC:cn	Appendix A # 3.2.2
CUTYPE	IC:cutype	Appendix A # 3.2.3
DELEGATED-FROM	IC:delegated-from	Appendix A # 3.2.4
DELEGATED-TO	IC:delegated-to	Appendix A # 3.2.5
DIR	IC:dir	Appendix A # 3.2.6
ENCODING	IC:encoding	Appendix A # 3.2.7
FMTTYPE	IC:fmttype	Appendix A # 3.2.8
FBTYPE	IC:fbtype	Appendix A # 3.2.9
LANGUAGE	IC:language	Appendix A # 3.2.10
MEMBER	IC:member	Appendix A # 3.2.11
PARTSTAT	IC:partstat	Appendix A # 3.2.12
RANGE	IC:range	Appendix A # 3.2.13
RELATED	IC:related	Appendix A # 3.2.14
RELTYPE	IC:reltype	Appendix A # 3.2.15
ROLE	IC:role	Appendix A # 3.2.16
RSVP	IC:rsvp	Appendix A # 3.2.17
SENT-BY	IC:sent-by	Appendix A # 3.2.18
TZID	IC:tzid	Appendix A # 3.2.19

3.5.1. VALUE parameter

iCalendar defines a "VALUE" property parameter (Section 3.2.20 of [RFC5545]). This property parameter is not mapped to an xCal XML element. Instead, the value type is handled by having different XML elements for each value, and these appear inside of IC:property elements. Thus, when converting from iCalendar to xCal, any "VALUE" property parameters are skipped. When converting from xCal into iCalendar, the appropriate "VALUE" property parameter MUST be included in the iCalendar property if the value type is not the default value type for that property.

3.6. Values (RFC5545 section 3.3)

In the typical case, iCalendar value types are mapped into XML elements with a matching name in all lowercase. In the case of the value for a recurrence rule (see below), iCalendar defines "structured" values and these are mapped into separate child elements for each value element.

3.6.1. Binary (RFC5545 section 3.3.1)

Description: iCalendar "BINARY" property values are represented by the IC:binary XML element. The content of the element is base64 encoded data. Whitespace MAY be inserted into the data at any point to "wrap" the data to reasonable line lengths. When converting back to iCalendar the whitespace MUST first be removed.

XML Definition: Appendix A # 3.3.1

Example:

<binary>SGVsbG8gV29ybGQh</binary>

3.6.2. Boolean (RFC5545 section 3.3.2)

Description: iCalendar "BOOLEAN" property values are represented by the IC:boolean XML element. The content of the element is a boolean value.

XML Definition: Appendix A # 3.3.2

Example:

<boolean>true</boolean>

3.6.3. Calendar User Address (RFC5545 section 3.3.3)

Description: iCalendar "CAL-ADDRESS" property values are represented by the IC:cal-address XML element. The content of the element is a URI.

XML Definition: Appendix A # 3.3.3

Example:

<cal-address>mailto:cyrus@example.com</cal-address>

3.6.4. Date (RFC5545 section 3.3.4)

Description: iCalendar "DATE" property values are represented by the IC:date XML element. The content of the element is the same date value specified by [RFC5545], with the exception that the date components are separated by "-" characters.

XML Definition: Appendix A # 3.3.4

Example:

```
<date>2011-05-17</date>
```

3.6.5. Date-Time (RFC5545 section 3.3.5)

Description: iCalendar "DATE-TIME" property values are represented by the IC:date-time XML element. The content of the element is the same date-time value specified by [RFC5545], with the exception that the date components are separated by "-" characters, and the time components are separated by ":" characters.

XML Definition: Appendix A # 3.3.5

Example:

```
<date-time>2011-05-17T12:00:00</date-time>
```

3.6.6. Duration (RFC5545 section 3.3.6)

Description: iCalendar "DURATION" property values are represented by the IC:duration XML element. The content of the element is the same duration value specified by [RFC5545].

XML Definition: Appendix A # 3.3.6

Example:

```
<duration>P1D</duration>
```

3.6.7. Float (RFC5545 section 3.3.7)

Description: iCalendar "FLOAT" property values are represented by the IC:float XML element. The content of the element is a text representation of a floating point number.

XML Definition: Appendix A # 3.3.7

Example:

```
<float>0.5</float>
```

3.6.8. Integer (RFC5545 section 3.3.8)

Description: iCalendar "INTEGER" property values are represented by the IC:integer XML element. The content of the element is a text representation of an integer number.

XML Definition: Appendix A # 3.3.8

Example:

```
<integer>50</integer>
```

3.6.9. Period of Time (RFC5545 section 3.3.9)

Description: iCalendar "PERIOD" property values are represented by the IC:period XML element. The content of the element is child elements representing the start, end or duration components of the period.

XML Definition: Appendix A # 3.3.9

Example:

```
<period>
  <start>2011-05-17T12:00:00</start>
  <duration>P1H</duration>
</period>
```

3.6.10. Recurrence Rule (RFC5545 section 3.3.10)

Description: iCalendar "RECUR" property values are represented by the IC:recur XML element. The content of the element is child elements representing the various components of a recurrence rule.

XML Definition: Appendix A # 3.3.10

Example:

```
<recur>
  <freq>YEARLY</freq>
  <count>5</count>
  <byday>-1SU</byday>
  <bymonth>10</bymonth>
</recur>
```

3.6.11. Text (RFC5545 section 3.3.11)

Description: iCalendar "TEXT" property values are represented by the IC:text XML element. The content of the element is simple text.

XML Definition: Appendix A # 3.3.11

Example:

```
<text>Hello World!</text>
```

3.6.12. Time (RFC5545 section 3.3.12)

Description: iCalendar "TIME" property values are represented by the IC:time XML element. The content of the element is the same time value specified by [RFC5545], with the exception that the time components are separated by ":" characters.

XML Definition: Appendix A # 3.3.12

Example:

```
<time>12:00:00</time>
```

3.6.13. URI (RFC5545 section 3.3.13)

Description: iCalendar "URI" property values are represented by the IC:uri XML element. The content of the element is a URI.

XML Definition: Appendix A # 3.3.13

Example:

```
<uri>http://calendar.example.com</uri>
```

3.6.14. UTC Offset (RFC5545 section 3.3.14)

Description: iCalendar "UTC-OFFSET" property values are represented by the IC:utc-offset XML element. The content of the element is the same UTC offset value specified by [RFC5545], with the exception that the hour and minute components are separated by a ":" character.

XML Definition: Appendix A # 3.3.14

Example:

```
<utc-offset>-05:00</utc-offset>
```

3.7. Extensions

iCalendar extension properties and property parameters (those with an "X-" prefix in their name) are handled in the same way as other properties and property parameters: the property or property parameter is represented by an XML element with the same name, but in lowercase. e.g., the "X-FOO" property in iCalendar turns into the IC: x-foo element in xCal. But see Section 5 for how to deal with default values for unrecognized extension properties or property parameters.

4. Converting from xCal into iCalendar

When converting component, property and property parameter values, the names SHOULD be converted to uppercase. Although iCalendar names are case insensitive, common practice is to keep them all uppercase following the actual definitions in [RFC5545].

Backslash escaping and line folding MUST be applied to the resulting iCalendar data as required by [RFC5545].

Non-binary value types MUST NOT be base64 encoded.

4.1. Converting XML Extensions into iCalendar

XML extensions are converted back to iCalendar in one of two ways, depending on whether the extensions are in the iCalendar XML namespace, or in an external namespace.

Extensions that are part of the iCalendar XML namespace MUST have element names that begin with "x-", and will be converted back to the equivalent extension property in iCalendar. For example, the "x-foo" element will convert to the "X-FOO" iCalendar property.

Extensions that are in a namespace other than the iCalendar XML namespace SHOULD be preserved in the iCalendar representation using the "XML" iCalendar property described in Section 4.2.

4.2. The XML property for iCalendar

This section describes an extension property for iCalendar, as covered in section 8.2.3 of [RFC5545].

Property name: XML

Purpose: To embed XML-encoded calendar data in the iCalendar format.

Value type: The default value type is "TEXT". The value type can

also be set to "BINARY" to indicate base64 encoded content.

Property parameters: IANA, non-standard, inline encoding, and value data type property parameters can be specified on this property.

Conformance: The property can appear on any iCalendar component.

Description: The value of this property is an XML element. The "XML" property MUST NOT be used to contain properties that are already defined in iCalendar, or properties that use the "X-" iCalendar extension property syntax. Since all elements in the urn:ietf:params:xml:ns:icalendar-2.0 namespace convert to a well-defined iCalendar object, the elements in this property MUST NOT be in the urn:ietf:params:xml:ns:icalendar-2.0 namespace. The XML element which is the value of this property MUST have an XML namespace declaration.

Note that the source XML may contain characters not allowed in iCalendar such as control characters. If this is the case, then the XML data MUST be base64 encoded. As required by [RFC5545], the "ENCODING" property parameter MUST be present and set to "BASE64", and the "VALUE" property parameter MUST be present and set to "BINARY".

There can be more than one "XML" property present for a given iCalendar object. The ordering of "XML" properties is not preserved in the conversion between xCal and iCalendar.

Format definition: This property is defined by the following notation:

```
xml      = "XML" xmlparam ( ":" text ) /
               (
                 ";" "ENCODING" "=" "BASE64"
                 ";" "VALUE" "=" "BINARY"
                 ":" binary
               )
               CRLF

xmlparam     = *(";" other-param)
```

Example: The following is an example of an iCalendar event with a location embedded in KML markup inside the "XML" property.

```
BEGIN:VCALENDAR
CALSCALE:GREGORIAN
PRODID:-//Example Inc.//Example Calendar//EN
VERSION:2.0
BEGIN:VEVENT
DTSTAMP:20080205T191224Z
DTSTART:20081006
SUMMARY:Planning meeting
UID:4088E990AD89CB3DBB484909
XML:<kml xmlns="http://www.opengis.net/kml/2.2">...</kml>
END:VEVENT
END:VCALENDAR
```

5. Handling Unrecognized Properties or Parameters

In iCalendar, properties have a default value type specified by their definition, e.g. "SUMMARY"'s value type is "TEXT" and "DURATION"'s is "DURATION". When a property uses its default value type, the "VALUE" property parameter does not need to be specified on the property.

When new properties are defined or "X-" properties used, an iCalendar<->xCal converter might not recognize them, and know what the appropriate default value types are, yet they need to be able to preserve the values. A similar issue arises for unrecognized property parameters. As a result, the following rules are applied when dealing with unrecognized properties and property parameters:

- When converting iCalendar into xCal:
  - Any property that does not include a "VALUE" property parameter and whose default value type is not known, MUST be converted using the value type XML element IC:unknown. The content of that element is the unprocessed value text.
  - Any unrecognized property parameter MUST be converted using the value type XML element IC:unknown, with its content set to the property parameter value text, treated as if it were a "TEXT" value, or list of "TEXT" values.
- When converting xCal into iCalendar:
  - Any IC:unknown property value XML elements are converted directly into iCalendar values. The containing property MUST NOT have a "VALUE" property parameter.
  - Any IC:unknown parameter value XML elements are converted as if they were IC:text value type XML elements.

Example: The following is an example of an unrecognized iCalendar property (that uses a "DATE-TIME" value as its default), and the equivalent xCal representation of that property.

iCalendar:

```
X-PROPERTY:20110512T120000Z
```

xCal:

```
<x-property>
  <unknown>20110512T120000Z</unknown>
</x-property>
```

Example: The following is an example of an unrecognized iCalendar property parameter (that uses a "DURATION" value as its default) specified on a recognized iCalendar property, and the equivalent xCal representation of that property and property parameter.

iCalendar:

```
DTSTART;X-PARAM=PT30M:20110512T130000Z
```

xCal:

```
<dtstart>
  <parameters>
    <x-param><unknown>PT30M</unknown></x-param>
  </parameters>
  <date-time>2011-05-12T13:00:00Z</date-time>
</dtstart>
```

6. Security Considerations

For security considerations specific to calendar data, see Section 7 of [RFC5545]. Since this specification is a mapping from iCalendar, no new security concerns are introduced related to calendar data.

The use of XML as a format does have security risks. Section 7 of [RFC3470] discusses these risks. See also the security discussion for the application/xml type in [RFC3023].

7. IANA Considerations

This document defines a new URN to identify a new XML namespace for iCalendar data. The URN conforms to a registry mechanism described in [RFC3688].

This document defines a new media type. The registration is in Section 7.2.

This document defines a new property for iCalendar. The registration is in Section 7.3.

7.1. Namespace Registration

Registration request for the iCalendar namespace:

URI: urn:ietf:params:xml:ns:icalendar-2.0

Registrant Contact: See the "Authors' Addresses" section of this document.

XML: None. Namespace URIs do not represent an XML specification.

7.2. Media Type

This section defines the MIME media type for use with iCalendar in XML data.

Type name: application

Subtype name: calendar+xml

Required parameters: none

Optional parameters: method, component and optinfo as defined for the text/calendar media type in [RFC5545]; charset as defined for application/xml in [RFC3023]; per [RFC3023], use of the charset property parameter with the value "utf-8" is "STRONGLY RECOMMENDED"

Encoding considerations: Same as encoding considerations of application/xml as specified in [RFC3023]

Security considerations: See Section 6.

Interoperability considerations: This media type provides an alternative format for iCalendar data based on XML.

Published specification: This specification.

Applications which use this media type: Applications that currently make use of the text/calendar media type can use this as an alternative.

Additional information:

Magic number(s): None

File extension(s): xcs

Macintosh file type code(s): None specified.

Person & email address to contact for further information:
calsify@ietf.org

Intended usage: COMMON

Restrictions on usage: There are no restrictions on where this media type can be used.

Author: See the "Author's Address" section of this document.

Change controller: IETF

7.3. iCalendar Property Registrations

This document defines the following new iCalendar property to be added to the registry defined in Section 8.2.3 of [RFC5545]:

Property	Status	Reference
XML	Current	RFCXXXX, Section 4.2

8. Acknowledgments

The authors would like to thank the following for their valuable contributions: Toby Considine, Bernard Desruisseaux, Keith Moore, Filip Navara, Simon Perreault, Arnaud Quillaud, Peter Saint-Andre, and Dave Thewlis. This specification originated from the work of the XML technical committee of the Calendaring and Scheduling Consortium.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3023] Murata, M., St. Laurent, S., and D.

Kohn, "XML Media Types", RFC 3023, January 2001.

[RFC3470] Hollenbeck, S., Rose, M., and L. Masinter, "Guidelines for the Use of Extensible Markup Language (XML) within IETF Protocols", BCP 70, RFC 3470, January 2003.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[RFC5545] Desruisseaux, B., "Internet Calendaring and Scheduling Core Object Specification (iCalendar)", RFC 5545, September 2009.

9.2. Informative References

[W3C.REC-xmlschema-2-20041028] Biron, P. and A. Malhotra, "XML Schema Part 2: Datatypes Second Edition", World Wide Web Consortium Recommendation REC-xmlschema-2-20041028, October 2004, <http://www.w3.org/TR/2004/REC-xmlschema-2-20041028>.

Appendix A. Relax NG Schema

Below is a Relax NG schema for iCalendar in XML. The schema is non-normative and given for reference only.

This schema uses the compact notation of Relax NG. The numeric section numbers given in the comments refer to section in [RFC5545]. The ordering of elements follows the section ordering of [RFC5545].

The Relax NG compact notation "?" operator is used to indicate an unordered list of items. However, that operator, as defined, allows "mixing" each element that it operates on at any depth within the other elements, rather than just allowing "mixing" of siblings only. As a result, the schema provided allows certain constructs that are not allowed in iCalendar. Given that there is no sibling-only unordered list operator in RelaxNG, this is the best representation that can be given.

Patterns for date/time, duration and utc-offset values are given because those differ from the values used in iCalendar. More

```
   restrictive schema with patterns and numerical limits could be
   derived from the example schema here if more comprehensive schema
   validation is required.

   # Relax NG Schema for iCalendar in XML

   default namespace = "urn:ietf:params:xml:ns:icalendar-2.0"

   # 3.2 Property Parameters

   # 3.2.1 Alternate Text Representation

   altrepparam = element altrep {
       value-uri
   }

   # 3.2.2 Common Name

   cnparam = element cn {
       value-text
   }

   # 3.2.3 Calendar User Type

   cutypeparam = element cutype {
       element text {
           "INDIVIDUAL" |
           "GROUP" |
           "RESOURCE" |
           "ROOM" |
           "UNKNOWN"
       }
   }

   # 3.2.4 Delegators

   delfromparam = element delegated-from {
       value-cal-address+
   }

   # 3.2.5 Delegatees

   deltoparam = element delegated-to {
       value-cal-address+
   }

   # 3.2.6 Directory Entry Reference
```

```
   dirparam = element dir {
       value-uri
   }

   # 3.2.7 Inline Encoding

   encodingparam = element encoding {
       element text {
           "8BIT" |
           "BASE64"
       }
   }


   # 3.2.8 Format Type

   fmttypeparam = element fmttype {
       value-text
   }

   # 3.2.9 Free/Busy Time Type

   fbtypeparam = element fbtype {
       element text {
           "FREE" |
           "BUSY" |
           "BUSY-UNAVAILABLE" |
           "BUSY-TENTATIVE"
       }
   }

   # 3.2.10 Language

   languageparam = element language {
       value-text
   }

   # 3.2.11 Group or List Membership

   memberparam = element member {
       value-cal-address+
   }

   # 3.2.12 Participation Status

   partstatparam = element partstat {
       type-partstat-event |
       type-partstat-todo |
```

```
        type-partstat-jour
    }

    type-partstat-event = (
        element text {
            "NEEDS-ACTION" |
            "ACCEPTED" |
            "DECLINED" |
            "TENTATIVE" |
            "DELEGATED"
        }
    )

    type-partstat-todo = (
        element text {
            "NEEDS-ACTION" |
            "ACCEPTED" |
            "DECLINED" |
            "TENTATIVE" |
            "DELEGATED" |
            "COMPLETED" |
            "IN-PROCESS"
        }
    )

    type-partstat-jour = (
        element text {
            "NEEDS-ACTION" |
            "ACCEPTED" |
            "DECLINED"
        }
    )

    # 3.2.13 Recurrence Identifier Range

    rangeparam = element range {
        element text {
            "THISANDFUTURE"
        }
    }

    # 3.2.14 Alarm Trigger Relationship

    trigrelparam = element related {
        element text {
            "START" |
            "END"
        }
```

```
   }

   # 3.2.15 Relationship Type

   reltypeparam = element reltype {
       element text {
           "PARENT" |
           "CHILD" |
           "SIBLING"
       }
   }

   # 3.2.16 Participation Role

   roleparam = element role {
       element text {
           "CHAIR" |
           "REQ-PARTICIPANT" |
           "OPT-PARTICIPANT" |
           "NON-PARTICIPANT"
       }
   }

   # 3.2.17 RSVP Expectation

   rsvpparam = element rsvp {
       value-boolean
   }

   # 3.2.18 Sent By

   sentbyparam = element sent-by {
       value-cal-address
   }

   # 3.2.19 Time Zone Identifier

   tzidparam = element tzid {
       value-text
   }

   # 3.3 Property Value Data Types

   # 3.3.1 BINARY

   value-binary =  element binary {
       xsd:string
   }
```

```
   # 3.3.2 BOOLEAN

   value-boolean = element boolean {
       xsd:boolean
   }

   # 3.3.3 CAL-ADDRESS

   value-cal-address = element cal-address {
       xsd:anyURI
   }

   # 3.3.4 DATE

   pattern-date = xsd:string {
       pattern = "\d\d\d\d-\d\d-\d\d"
   }

   value-date = element date {
       pattern-date
   }

   # 3.3.5 DATE-TIME

   pattern-date-time = xsd:string {
       pattern = "\d\d\d\d-\d\d-\d\dT\d\d-\d\d-\d\dZ?"
   }

   value-date-time = element date-time {
       pattern-date-time
   }

   # 3.3.6 DURATION

   pattern-duration = xsd:string {
       pattern = "[+\-]?P(\d+W)|(\d+D)?"
               ~ "(T(\d+H(\d+M)?(\d+S)?)|"
               ~   "(\d+M(\d+S)?)|"
               ~   "(\d+S))?"
   }

   value-duration = element duration {
       pattern-duration
   }

   # 3.3.7 FLOAT

   value-float = element float {
```

```
        xsd:float
    }

    # 3.3.8 INTEGER

    value-integer = element integer {
        xsd:integer
    }

    # 3.3.9 PERIOD

    value-period = element period {
        element start {
            pattern-date-time
        },
        (
            element end {
                pattern-date-time
            } |
            element duration {
                pattern-duration
            }
        )
    }

    # 3.3.10 RECUR

    value-recur = element recur {
        type-freq,
        (type-until | type-count)?,
        element interval {
            xsd:positiveInteger
        }?,
        type-bysecond*,
        type-byminute*,
        type-byhour*,
        type-byday*,
        type-bymonthday*,
        type-byyearday*,
        type-byweekno*,
        type-bymonth*,
        type-bysetpos*,
        element wkst { type-weekday }?
    }

    type-freq = element freq {
        "SECONDLY" |
        "MINUTELY" |
```

```
        "HOURLY"   |
        "DAILY"    |
        "WEEKLY"   |
        "MONTHLY"  |
        "YEARLY"
    }

    type-until = element until {
        type-date |
        type-date-time
    }

    type-count = element count {
        xsd:positiveInteger
    }

    type-bysecond = element bysecond {
        xsd:positiveInteger
    }

    type-byminute = element byminute {
        xsd:positiveInteger
    }

    type-byhour = element byhour {
        xsd:positiveInteger
    }

    type-weekday = (
        "SU" |
        "MO" |
        "TU" |
        "WE" |
        "TH" |
        "FR" |
        "SA"
    )

    type-byday = element byday {
        xsd:integer?,
        type-weekday
    }

    type-bymonthday = element bymonthday {
        xsd:integer
    }

    type-byyearday = element byyearday {
```

```
        xsd:integer
    }

    type-byweekno = element byweekno {
        xsd:integer
    }

    type-bymonth = element bymonth {
        xsd:positiveInteger
    }

    type-bysetpos = element bysetpos {
        xsd:integer
    }

    # 3.3.11 TEXT

    value-text = element text {
        xsd:string
    }

    # 3.3.12 TIME

    pattern-time = xsd:string {
        pattern = "\d\d:\d\d:\d\dZ?"
    }

    value-time = element time {
        pattern-time
    }

    # 3.3.13 URI

    value-uri = element uri {
        xsd:anyURI
    }

    # 3.3.14 UTC-OFFSET

    value-utc-offset = element utc-offset {
        xsd:string { pattern = "[+\-]\d\d:\d\d" }
    }

    # UNKNOWN

    value-unknown = element unknown {
        xsd:string
    }
```

```
   # 3.4 iCalendar Stream

   start = element icalendar {
       vcalendar+
   }

   # 3.6 Calendar Components

   vcalendar = element vcalendar {
       type-calprops,
       type-component
   }

   type-calprops = element properties {
       property-prodid &
       property-version &
       property-calscale? &
       property-method?
   }

   type-component = element components {
       (
           component-vevent |
           component-vtodo |
           component-vjournal |
           component-vfreebusy |
           component-vtimezone
       )*
   }

   # 3.6.1 Event Component

   component-vevent = element vevent {
       type-eventprop,
       element components {
           component-valarm+
       }?
   }

   type-eventprop = element properties {
       property-dtstamp &
       property-dtstart &
       property-uid &

       property-class? &
       property-created? &
       property-description? &
       property-geo? &
```

```
        property-last-mod? &
        property-location? &
        property-organizer? &
        property-priority? &
        property-seq? &
        property-status-event? &
        property-summary? &
        property-transp? &
        property-url? &
        property-recurid? &

        property-rrule? &

        (property-dtend | property-duration)? &

        property-attach* &
        property-attendee* &
        property-categories* &
        property-comment* &
        property-contact* &
        property-exdate* &
        property-rstatus* &
        property-related* &
        property-resources* &
        property-rdate*
    }

    # 3.6.2 To-do Component

    component-vtodo = element vtodo {
        type-todoprop,
        element components {
            component-valarm+
        }?
    }

    type-todoprop = element properties {
        property-dtstamp &
        property-uid &

        property-class? &
        property-completed? &
        property-created? &
        property-description? &
        property-geo? &
        property-last-mod? &
        property-location? &
        property-organizer? &
```

```
        property-percent? &
        property-priority? &
        property-recurid? &
        property-seq? &
        property-status-todo? &
        property-summary? &
        property-url? &

        property-rrule? &

        (
            (property-dtstart?, property-dtend? ) |
            (property-dtstart, property-duration)?
        ) &

        property-attach* &
        property-attendee* &
        property-categories* &
        property-comment* &
        property-contact* &
        property-exdate* &
        property-rstatus* &
        property-related* &
        property-resources* &
        property-rdate*
    }

    # 3.6.3 Journal Component

    component-vjournal = element vjournal {
        type-jourprop
    }

    type-jourprop = element properties {
        property-dtstamp &
        property-uid &

        property-class? &
        property-created? &
        property-dtstart? &
        property-last-mod? &
        property-organizer? &
        property-recurid? &
        property-seq? &
        property-status-jour? &
        property-summary? &
        property-url? &
```

```
        property-rrule? &

        property-attach* &
        property-attendee* &
        property-categories* &
        property-comment* &
        property-contact* &
        property-description? &
        property-exdate* &
        property-related* &
        property-rdate* &
        property-rstatus*
    }

    # 3.6.4 Free/Busy Component

    component-vfreebusy = element vfreebusy {
        type-fbprop
    }

    type-fbprop = element properties {
        property-dtstamp &
        property-uid &

        property-contact? &
        property-dtstart? &
        property-dtend? &
        property-duration? &
        property-organizer? &
        property-url? &

        property-attendee* &
        property-comment* &
        property-freebusy* &
        property-rstatus*
    }

    # 3.6.5 Time Zone Component

    component-vtimezone = element vtimezone {
        element properties {
            property-tzid &

            property-last-mod? &
            property-tzuurl?
        },
        element components {
            (component-standard | component-daylight) &
```

```
            component-standard* &
            component-daylight*
        }
    }

    component-standard = element standard {
        type-tzprop
    }

    component-daylight = element daylight {
        type-tzprop
    }

    type-tzprop = element properties {
        property-dtstart &
        property-tzoffsetto &
        property-tzoffsetfrom &

        property-rrule? &

        property-comment* &
        property-rdate* &
        property-tzname*
    }

    # 3.6.6 Alarm Component

    component-valarm = element valarm {
        audioprop | dispprop | emailprop
    }

    type-audioprop = element properties {
        property-action &
        property-trigger &

        (property-duration, property-repeat)? &

        property-attach?
    }

    type-dispprop = element properties {
        property-action &
        property-description &
        property-trigger &
        property-summary &

        property-attendee+ &
```

```
        (property-duration, property-repeat)? &

        property-attach*
    }

    type-emailprop = element properties {
        property-action &
        property-description &
        property-trigger &

        (property-duration, property-repeat)?
    }

    # 3.7 Calendar Properties

    # 3.7.1 Calendar Scale

    property-calscale = element calscale {

        element parameters { empty }?,

        element text { "GREGORIAN" }
    }

    # 3.7.2 Method

    property-method = element method {

        element parameters { empty }?,

        value-text
    }

    # 3.7.3 Product Identifier

    property-prodid = element prodid {

        element parameters { empty }?,

        value-text
    }

    # 3.7.4 Version

    property-version = element version {

        element parameters { empty }?,
```

```
        element text { "2.0" }
    }

    # 3.8 Component Properties

    # 3.8.1 Descriptive Component Properties

    # 3.8.1.1 Attachment

    property-attach = element attach {

        element parameters {
            fmttypeparam? &
            encodingparam?
        }?,

        value-uri | value-binary
    }

    # 3.8.1.2 Categories

    property-categories = element categories {

        element parameters {
            languageparam? &
        }?,

        value-text+
    }

    # 3.8.1.3 Classification

    property-class = element class {

        element parameters { empty }?,

        element text {
            "PUBLIC" |
            "PRIVATE" |
            "CONFIDENTIAL"
        }
    }

    # 3.8.1.4 Comment

    property-comment = element comment {

        element parameters {
```

```
            altrepparam? &
            languageparam?
        }?,

        value-text
    }

    # 3.8.1.5 Description

    property-description = element description {

        element parameters {
            altrepparam? &
            languageparam?
        }?,

        value-text
    }

    # 3.8.1.6 Geographic Position

    property-geo = element geo {

        element parameters { empty }?,

        element latitude  { xsd:float },
        element longitude { xsd:float }
    }

    # 3.8.1.7 Location

    property-location = element location {

        element parameters {
            altrepparam? &
            languageparam?
        }?,

        value-text
    }

    # 3.8.1.8 Percent Complete

    property-percent = element percent-complete {

        element parameters { empty }?,

        value-integer
```

```
   }

   # 3.8.1.9 Priority

   property-priority = element priority {

       element parameters { empty }?,

       value-integer
   }

   # 3.8.1.10 Resources

   property-resources = element resources {

       element parameters {
           altrepparam? &
           languageparam?
       }?,

       value-text+
   }

   # 3.8.1.11 Status

   property-status-event = element status {

       element parameters { empty }?,

       element text {
           "TENTATIVE" |
           "CONFIRMED" |
           "CANCELLED"
       }
   }

   property-status-todo = element status {

       element parameters { empty }?,

       element text {
           "NEEDS-ACTION" |
           "COMPLETED" |
           "IN-PROCESS" |
           "CANCELLED"
       }
   }
```

```
   property-status-jour = element status {

       element parameters { empty }?,

       element text {
           "DRAFT" |
           "FINAL" |
           "CANCELLED"
       }
   }

   # 3.8.1.12 Summary

   property-summary = element summary {

       element parameters {
           altrepparam? &
           languageparam?
       }?,

       value-text
   }

   # 3.8.2 Date and Time Component Properties

   # 3.8.2.1 Date/Time Completed

   property-completed = element completed {

       element parameters { empty }?,

       value-date-time
   }

   # 3.8.2.2 Date/Time End

   property-dtend = element dtend {

       element parameters {
           tzidparam?
       }?,

       value-date-time |
       value-date
   }

   # 3.8.2.3 Date/Time Due
```

```
   property-due = element due {

       element parameters {
           tzidparam?
       }?,

       value-date-time |
       value-date
   }

   # 3.8.2.4 Date/Time Start

   property-dtstart = element dtstart {

       element parameters {
           tzidparam?
       }?,

       value-date-time |
       value-date
   }

   # 3.8.2.5 Duration

   property-duration = element duration {

       element parameters { empty }?,

       value-duration
   }

   # 3.8.2.6 Free/Busy Time

   property-freebusy = element freebusy {

       element parameters {
           fbtypeparam?
       }?,


       value-period+
   }

   # 3.8.2.7 Time Transparency

   property-transp = element transp {

       element parameters { empty }?,
```

```
        element text {
            "OPAQUE" |
            "TRANSPARENT"
        }
    }

    # 3.8.3 Time Zone Component Properties

    # 3.8.3.1 Time Zone Identifier

    property-tzid = element tzid {

        element parameters { empty }?,

        value-text
    }

    # 3.8.3.2 Time Zone Name

    property-tzname = element tzname {

        element parameters {
            languageparam?
        }?,

        value-text
    }

    # 3.8.3.3 Time Zone Offset From

    property-tzoffsetfrom = element tzoffsetfrom {

        element parameters { empty }?,

        value-utc-offset
    }

    # 3.8.3.4 Time Zone Offset To

    property-tzoffsetto = element tzoffsetto {

        element parameters { empty }?,

        value-utc-offset
    }

    # 3.8.3.5 Time Zone URL
```

```
   property-tzurl = element tzurl {

       element parameters { empty }?,

       value-uri
   }

   # 3.8.4 Relationship Component Properties

   # 3.8.4.1 Attendee

   property-attendee = element attendee {

       element parameters {
           cutypeparam? &
           memberparam? &
           roleparam? &
           partstatparam? &
           rsvpparam? &
           deltoparam? &
           delfromparam? &
           sentbyparam? &
           cnparam? &
           dirparam? &
           languageparam?
       }?,

       value-cal-address
   }

   # 3.8.4.2 Contact

   property-contact = element contact {

       element parameters {
           altrepparam? &
           languageparam?
       }?,

       value-text
   }

   # 3.8.4.3 Organizer

   property-organizer = element organizer {

       element parameters {
           cnparam? &
```

```
            dirparam? &
            sentbyparam? &
            languageparam?
        }?,

        value-cal-address
    }

    # 3.8.4.4 Recurrence ID

    property-recurid = element recurrence-id {

        element parameters {
            tzidparam? &
            rangeparam?
        }?,

        value-date-time |
        value-date
    }

    # 3.8.4.5 Related-To

    property-related = element related-to {

        element parameters {
            reltypeparam?
        }?,

        value-text
    }

    # 3.8.4.6 Uniform Resource Locator

    property-url = element url {

        element parameters { empty }?,

        value-uri
    }

    # 3.8.4.7 Unique Identifier

    property-uid = element uid {

        element parameters { empty }?,

        value-text
```

```
   }

   # 3.8.5 Recurrence Component Properties

   # 3.8.5.1 Exception Date/Times

   property-exdate = element exdate {

       element parameters {
           tzidparam?
       }?,

       value-date-time+ |
       value-date+
   }

   # 3.8.5.2 Recurrence Date/Times

   property-rdate = element rdate {

       element parameters {
           tzidparam?
       }?,

       value-date-time+ |
       value-date+ |
       value-period+
   }

   # 3.8.5.3 Recurrence Rule

   property-rrule = element rrule {

       element parameters { empty }?,

       value-recur
   }

   # 3.8.6 Alarm Component Properties

   # 3.8.6.1 Action

   property-action = element action {

       element parameters { empty }?,

       element text {
           "AUDIO" |
```

```
            "DISPLAY" |
            "EMAIL"
        }
    }

    # 3.8.6.2 Repeat Count

    property-repeat = element repeat {

        element parameters { empty }?,

        value-integer
    }

    # 3.8.6.3 Trigger

    property-trigger = element trigger {

        (
            element parameters {
                trigrelparam?
            }?,

            value-duration
        ) |
        (
            element parameters { empty }?,

            value-date-time
        )
    }

    # 3.8.7 Change Management Component Properties

    # 3.8.7.1 Date/Time Created

    property-created = element created {

        element parameters { empty }?,

        value-date-time
    }

    # 3.8.7.2 Date/Time Stamp

    property-dtstamp = element dtstamp {

        element parameters { empty }?,
```

```
        value-date-time
    }

    # 3.8.7.3 Last Modified

    property-last-mod = element last-modified {

        element parameters { empty }?,

        value-date-time
    }

    # 3.8.7.4 Sequence Number

    property-seq = element sequence {

        element parameters { empty }?,

        value-integer
    }

    # 3.8.8 Miscellaneous Component Properties

    # 3.8.8.3 Request Status

    property-rstatus = element request-status {

        element parameters {
            languageparam?
        }?,

        element code { xsd:string },
        element description { xsd:string },
        element data { xsd:string }?
    }
```

Appendix B. Examples

This section contains two examples of iCalendar objects with their xCal representation.

B.1. Example 1

B.1.1. iCalendar Data

```
BEGIN:VCALENDAR
CALSCALE:GREGORIAN
PRODID:-//Example Inc.//Example Calendar//EN
VERSION:2.0
BEGIN:VEVENT
DTSTAMP:20080205T191224Z
DTSTART:20081006
SUMMARY:Planning meeting
UID:4088E990AD89CB3DBB484909
END:VEVENT
END:VCALENDAR
```

B.1.2. XML Data

```
<?xml version="1.0" encoding="utf-8"?>
<icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
 <vcalendar>
  <properties>
   <calscale>
     <text>GREGORIAN</text>
   </calscale>
   <prodid>
    <text>-//Example Inc.//Example Calendar//EN</text>
   </prodid>
   <version>
     <text>2.0</text>
   </version>
  </properties>
  <components>
   <vevent>
    <properties>
     <dtstamp>
       <date-time>2008-02-05T19:12:24Z</date-time>
     </dtstamp>
     <dtstart>
       <date>2008-10-06</date>
     </dtstart>
     <summary>
      <text>Planning meeting</text>
     </summary>
     <uid>
      <text>4088E990AD89CB3DBB484909</text>
     </uid>
    </properties>
   </vevent>
  </components>
 </vcalendar>
</icalendar>
```

B.2. Example 2

B.2.1. iCalendar Data

```
VERSION:2.0
PRODID:-//Example Corp.//Example Client//EN
BEGIN:VTIMEZONE
LAST-MODIFIED:20040110T032845Z
TZID:US/Eastern
BEGIN:DAYLIGHT
DTSTART:20000404T020000
RRULE:FREQ=YEARLY;BYDAY=1SU;BYMONTH=4
TZNAME:EDT
TZOFFSETFROM:-0500
TZOFFSETTO:-0400
END:DAYLIGHT
BEGIN:STANDARD
DTSTART:20001026T020000
RRULE:FREQ=YEARLY;BYDAY=-1SU;BYMONTH=10
TZNAME:EST
TZOFFSETFROM:-0400
TZOFFSETTO:-0500
END:STANDARD
END:VTIMEZONE
BEGIN:VEVENT
DTSTAMP:20060206T001121Z
DTSTART;TZID=US/Eastern:20060102T120000
DURATION:PT1H
RRULE:FREQ=DAILY;COUNT=5
RDATE;TZID=US/Eastern;VALUE=PERIOD:20060102T150000/PT2H
SUMMARY:Event #2
DESCRIPTION:We are having a meeting all this week at 12 pm fo
 r one hour\, with an additional meeting on the first day 2 h
 ours long.\nPlease bring your own lunch for the 12 pm meetin
 gs.
UID:00959BC664CA650E933C892C@example.com
END:VEVENT
BEGIN:VEVENT
DTSTAMP:20060206T001121Z
DTSTART;TZID=US/Eastern:20060104T140000
DURATION:PT1H
RECURRENCE-ID;TZID=US/Eastern:20060104T120000
SUMMARY:Event #2 bis
UID:00959BC664CA650E933C892C@example.com
END:VEVENT
END:VCALENDAR
```

B.2.2. XML Data

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
   <icalendar xmlns="urn:ietf:params:xml:ns:icalendar-2.0">
     <vcalendar>
       <properties>
         <prodid>
           <text>-//Example Inc.//Example Client//EN</text>
         </prodid>
         <version>
           <text>2.0</text>
         </version>
       </properties>
       <components>
         <vtimezone>
           <properties>
             <last-modified>
               <date-time>2004-01-10T03:28:45Z</date-time>
             </last-modified>
             <tzid>US/Eastern</tzid>
           </properties>
           <components>
             <daylight>
               <properties>
                 <dtstart>
                   <date-time>2000-04-04T02:00:00</date-time>
                 </dtstart>
                 <rrule>
                   <recur>
                     <freq>YEARLY</freq>
                     <byday>1SU</byday>
                     <bymonth>4</bymonth>
                   </recur>
                 </rrule>
                 <tzname>
                   <text>EDT</text>
                 </tzname>
                 <tzoffsetfrom>
                   <utc-offset>-05:00</utc-offset>
                 </tzoffsetfrom>
                 <tzoffsetto>
                   <utc-offset>-04:00</utc-offset>
                 </tzoffsetto>
               </properties>
             </daylight>
             <standard>
               <properties>
                 <dtstart>
                   <date-time>2000-10-26T02:00:00</date-time>
                 </dtstart>
                 <rrule>
```

```
                <recur>
                  <freq>YEARLY</freq>
                  <byday>-1SU</byday>
                  <bymonth>10</bymonth>
                </recur>
              </rrule>
              <tzname>
                <text>EST</text>
              </tzname>
              <tzoffsetfrom>
                <utc-offset>-04:00</utc-offset>
              </tzoffsetfrom>
              <tzoffsetto>
                <utc-offset>-05:00</utc-offset>
              </tzoffsetto>
            </properties>
          </standard>
        </components>
      </vtimezone>
      <vevent>
        <properties>
          <dtstamp>
            <date-time>2006-02-06T00:11:21Z<date-time>
          </dtstamp>
          <dtstart>
            <parameters>
              <tzid><text>US/Eastern</text></tzid>
            </parameters>
            <date-time>2006-01-02T12:00:00</date-time>
          </dtstart>
          <duration>
            <duration>PT1H</duration>
          </duration>
          <rrule>
            <recur>
              <freq>DAILY</freq>
              <count>5</count>
            </recur>
          </rrule>
          <rdate>
            <parameters>
              <tzid><text>US/Eastern</text></tzid>
            </parameters>
            <period>
              <start>2006-01-02T15:00:00</start>
              <duration>PT2H</duration>
            </period>
          </rdate>
```

```
            <summary>
              <text>Event #2</text>
            </summary>
            <description>
              <text>We are having a meeting all this week at 12
   pm for one hour, with an additional meeting on the first day
   2 hours long.&#x0a;Please bring your own lunch for the 12 pm
   meetings.</text>
            </description>
            <uid>
              <text>00959BC664CA650E933C892C@example.com</text>
            </uid>
          </properties>
        </vevent>
        <vevent>
          <properties>
            <dtstamp>
              <date-time>2006-02-06T00:11:21Z</date-time>
            </dtstamp>
            <dtstart>
              <parameters>
                <tzid><text>US/Eastern</text></tzid>
              </parameters>
              <date-time>2006-01-04T14:00:00</date-time>
            </dtstart>
            <duration>
              <duration>PT1H</duration>
            </duration>
            <recurrence-id>
              <parameters>
                <tzid><text>US/Eastern</text></tzid>
              </parameters>
              <date-time>2006-01-04T12:00:00</date-time>
            </recurrence-id>
            <summary>
              <text>Event #2 bis</text>
            </summary>
            <uid>
              <text>00959BC664CA650E933C892C@example.com</text>
            </uid>
          </properties>
        </vevent>
      </components>
    </vcalendar>
  </icalendar>
```

Appendix C. Change History (to be removed prior to publication as an RFC)

Changes in -09:

1. LC: Changed namespace prefix to IC:.

2. LC: Clarified what needs to happen with base64 encoding.

3. LC: Changed "XML" to "xCal" in various places.

4. LC: Schema now uses xsd types where appropriate.

5. LC: GEO value fixed to be two floats.

6. LC: Fix use of <text> within parameters.

7. LC: Add IC:unknown value element to deal with unrecognized properties or parameters.

8. LC: Added escaped/folded text example property.

9. Removed <value> wrapper element for GEO and REQUEST-STATUS.

10. Date/time, duration and utc-offset values defined in terms of patterns. Some now align with xsd patterns rather than iCalendar ones.

11. Period schema value defined in terms of child elements.

12. Added more examples.

Changes in -08:

1. Updated optional parameters section of media type registration (feedback from ietf-types)

2. Corrected the schema for the trigger element

3. Updated to ietf.org mailing list

4. Updated definition of iCalendar XML property

Changes in -07:

1. Updated XML reference to latest version.

2. Corrected dtstamp elements in samples.

3. Removed empty informative references section.

4. Changed media type registration file extension to "xcs" (feedback from ietf-types)

5. Changed media type registration follow up email to ietf-calsify (feedback from ietf-types)

6. Updated several parts of the specification to show how future extensions to iCalendar can be handled without changing this specification (feedback from ietf-types)

7. Updated security concerns in section 5 (feedback from ietf-types)

8. Updated encoding considerations in section 5 (feedback from ietf-types)

Changes in -06:

1. Removed the ordwk and weekday elements from the byday element in a recurrence rule. The byday element now takes a text value which is the same as the value used in [RFC5545].

2. Added another example of an iCalendar object converted to xCal.

3. Fixed the MIME type registration (another instance of xml+calendar corrected to calendar+xml).

Changes in -05:

1. Added description to XML extension discussing how to handle binary data in XML.

2. Removed empty Appendix B.

Changes in -04:

1. Changed the proposed MIME type from xml+calendar to calendar+xml.

2. Fixed several references to sections of RFC5545.

3. Updated example in Appendix C.

4. Corrected the definition and grammar for TIME and UTC-OFFSET properties.

Changes in -03:

1. Removed the LINK extension and related sections. The concept will be addressed in a separate specification.

2. Various minor edits for clarity and consistency.

Changes in -02:

1. Added LINK extension to iCalendar and section discussing links in XML format.

2. Adopted "xCal" as the short name for the specification.

Changes in -01:

1. Changed 2445bis references to RFC5545.

2. Added a version number to the XML namespace for iCalendar.

3. Changed the values for the date, date-time, period, and duration elements to exactly match the values specified in RFC5545. Previously these were broken out into separate elements for day, month, year, etc.

4. Added specification for XML property in iCalendar.

Authors' Addresses

Cyrus Daboo
Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
USA

EMail: cyrus@daboo.name
URI: http://www.apple.com/

Mike Douglass
Rensselaer Polytechnic Institute
110 8th Street
Troy, NY 12180
USA

EMail: douglm@rpi.edu
URI: http://www.rpi.edu/

Steven Lees
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
USA

EMail: steven.lees@microsoft.com
URI: http://www.microsoft.com/