

vObject — vObject Model and vFormat Syntax

Working Draft Standard

Warning for drafts

This document is not a CalConnect Standard. It is distributed for review and comment, and is subject to change without notice and may not be referred to as a Standard. Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

The Calendaring and Scheduling Consortium, Inc. 2016

© 2016 The Calendaring and Scheduling Consortium, Inc.

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from the address below.

The Calendaring and Scheduling Consortium, Inc.

4390 Chaffin Lane
McKinleyville
California 95519
United States of America

copyright@calconnect.org
www.calconnect.org

Contents

Introduction.....	v
1. Scope.....	1
2. Normative references.....	1
3. Overview.....	1
4. PATCH Component.....	2
5. iCalendar Path.....	2
6. Adding or Updating Components.....	5
7. Adding or Updating Properties.....	6
8. Deleting Components, Properties, or Property Parameters.....	7
9. Adding or Updating Property Parameters.....	7
10. iCalendar Extensions.....	8
11. VPATCH Component.....	8
11.1.PATCH Component.....	8
12. VPATCH Properties.....	9
12.1.PATCH-VERSION Property.....	9
12.2.PATCH-ORDER Property.....	10
13. PATCH Component Properties.....	11
13.1.PATCH-TARGET Property.....	11
13.2.PATCH-DELETE Property.....	11
13.3.PATCH-PARAMETER Property.....	12
13.4.PATCH-ACTION Property Parameter.....	12
14. Additional Considerations.....	13
14.1.Handling Default Properties and Parameters.....	13
14.2.Handling Recurrences.....	14
14.3.Folded lines.....	15
14.4.Encoding.....	16
14.5.Generation.....	16
15. Use with iTIP.....	16
16. Use with CalDAV and HTTP.....	16
17. Security Considerations.....	17
18. Privacy considerations.....	17
19. IANA Considerations.....	17
19.1.Component Registrations.....	17
19.2.Property Registrations.....	17
19.3.Parameter Registrations.....	18
19.4.Property and Parameter Value Registries.....	18
20. VPATCH Examples.....	18
20.1Add a new component.....	18
20.2Add a new VALARM component.....	19
20.3.Replace a component.....	19
20.4.Remove a component.....	19
20.5.Add properties to a component.....	20
20.6.Update properties in a component.....	20
20.7.Update a targeted property in a component.....	20
20.8.Remove a property from a component.....	21
20.9.Remove a property with a specific value from a component.....	21
20.10Change a parameter on a property with a specific value from a component.....	21
20.11Remove a parameter on a property with a specific value from a component.....	22
21. Remove a value from a multi-valued parameter on a property with a specific value from a component.....	22

21.1.Remove a value from a multi-valued property from a component.....	22
21.2.Attendee updating their participation status.....	23
21.3.Recurring event adding one override.....	23
21.4.Removal of an overridden instance.....	24
22. Acknowledgements.....	25
Appendix A Change History (To be removed by RFC Editor before publication).....	26
Bibliography.....	27

Introduction

The iCalendar [IETF RFC 5545](#) data format is in widespread use to represent calendar data. iCalendar data can grow large (e.g., a family calendar containing events over a period of several years). Updating large resources over the network currently requires the entire data to be sent—even if the change itself is minor.

This specification defines a new iCalendar component that can be used to “patch” (incrementally update) iCalendar data in an efficient manner. When combined with the HTTP PATCH method [IETF RFC 5789](#), it can be used to update calendar object resources on a CalDAV [IETF RFC 4791](#) server, or any resource on an HTTP server that contains iCalendar data.

vObject — vObject Model and vFormat Syntax

1. Scope

This document defines a new iCalendar [IETF RFC 5545](#) component that allows small “patches” to be applied to large iCalendar data objects, to allow for efficient data updates. It also describes how this new component can be used with the CalDAV calendar data access protocol [IETF RFC 4791](#).

2. Normative references

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

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

IETF RFC 5234, OVERELL, P. *Augmented BNF for Syntax Specifications: ABNF*. 2008. RFC Publisher. <https://www.rfc-editor.org/info/rfc5234>.

IETF RFC 5545, Internet Engineering Task Force (committee). *Internet Calendaring and Scheduling Core Object Specification (iCalendar)*. 2009. RFC Publisher. <https://www.rfc-editor.org/info/rfc5545>.

IETF RFC 5546, Internet Engineering Task Force (committee). *iCalendar Transport-Independent Interoperability Protocol (iTIP)*. 2009. RFC Publisher. <https://www.rfc-editor.org/info/rfc5546>.

IETF RFC 7953, DABOO, C., M. DOUGLASS and Internet Engineering Task Force. *Calendar Availability*. 2016. RFC Publisher. <https://www.rfc-editor.org/info/rfc7953>.

3. Overview

The basic design of the patch format is a “VPATCH” component (defined in Section 10.1) containing one or more “PATCH” components (defined in Section 10.1.1), each specifying a path (which identifies one or more components in the iCalendar object being patched), and other components and properties that define the set of changes to be made.

When multiple “VPATCH” components are present in an iCalendar object, the order in which they are applied is defined by the value of any “PATCH-ORDER” properties in the “VPATCH” components. The “VPATCH” components are sorted in order from lowest “PATCH-ORDER” integer value to highest, with any components not containing a “PATCH-ORDER” property placed last. The patch process is then applied in sorted order (any components with the same “PATCH-ORDER” value can be applied in any order).

No specific processing order is defined for multiple “PATCH” components in a “VPATCH” component.

The “VPATCH” component also contains an optional “PATCH-VERSION” property to allow future extensions to the format to be recognized. This document only defines version number “1”. The “PATCH-VERSION” property only needs to be present if the version number is greater than “1”. If a patch processing engine is unable to handle the indicated version it MUST reject the entire patch operation defined by the enclosing iCalendar object, even if other “VPATCH” components have a “PATCH-VERSION” number that is supported.

After applying a patch to an iCalendar object, the basic validity of the resulting iCalendar object SHOULD be checked by the processing engine (e.g., if the patch added an extra “DTSTART” property to a “VEVENT” component that would be considered a violation of [IETF RFC 5545](#)’s

cardinality rules for the “DTSTART” property in a “VEVENT”). If this occurs, the patch operation MUST fail.

Other validity constraints can be applied if needed. For example, CalDAV [IETF RFC 4791](#) requires that the “UID” property be the same in all components in a calendar object resource stored on the server. If a patch operation adds a component to an iCalendar object with a different “UID” value than the existing components, that result would be an invalid CalDAV calendar object resource. If other validity constraints are violated, the patch operation MUST fail.

Any failure to process a “VPATCH” component, for whatever reason, MUST result in the entire patch operation being cancelled, with the iCalendar object being patched left in its original state.

4. PATCH Component

A “PATCH” component (defined in Section 10.1.1) MUST contain one “PATCH-TARGET” property whose value is an iCalendar path (see Section 5) that identifies components within the iCalendar object being patched (see Section 11.2 for special handling of components representing recurring items). The set of components thus identified are the “target components” for the patch operations. The set of patch operations defined by the other components and properties present in the “PATCH” component are then applied to each target component (in the order specified below). If a “PATCH-TARGET” property does not match any components in the iCalendar object being patched, then the patch operation MUST succeed without any changes being applied to the iCalendar object being patched.

Four patch operations are supported:

- Component additions or updates: any components within the “PATCH” component are considered to be additions or updates (see Section 6).
- Property additions or updates: any properties (other than those whose name starts with the “PATCH-” prefix) are considered to be additions or updates (see Section 7).
- Component, property, property parameter, property value, or property parameter value deletion: indicated by the present of one or more “PATCH-DELETE” properties (see Section 8).
- Property parameter additions or updates: indicated by the presence of one or more “PATCH-PARAMETER” properties (see Section 9).

When processing a “PATCH” component, the processing engine MUST follow this order:

- Process all “PATCH-DELETE” properties first.
- Process all “PATCH-PARAMETER” properties second.
- Process all other components third.
- Process all non “PATCH-” prefixed properties fourth.

5. iCalendar Path

The “PATCH-TARGET”, “PATCH-DELETE”, and “PATCH-PARAMETER” property values are all iCalendar “paths”. The path is used to match iCalendar elements that the patch operation will be applied to. The path is a list of “segments” (separated by the “/”, “#”, “;” or “=” characters) that matches an iCalendar element in the iCalendar object model hierarchy (a component, a property, a property parameter, a property value, or a property parameter value). A path can either be “absolute” (referring to items within the top-level iCalendar object being patched) or “relative” (referring to items within some other component as determined by the scope of the operation).

A path can start with a series of component segments (which always have a “/” prefix). Those can be followed by a property segment (which always has a “#” prefix). A property segment can be followed by either a parameter segment (which always has a “;” prefix), or a value segment (which

always has a "=" prefix). A parameter segment can be followed by a value segment (which always has a "=" prefix).

An absolute path always starts with a "/VCALENDAR" component segment since an iCalendar object is always a single "VCALENDAR" component.

A relative path can start with a component segment or a property segment, with the path assumed to be relative to an enclosing component defined by the context.

To target a component inside of another component, a component segment is appended to the path. Component segments can include an optional match item. When present, this allows targeting of components that match a specific "UID" property value, and/or a "RECURRENCE-ID" value (or lack of "RECURRENCE-ID" property to target a "master" recurrence component). See Section 11.2 for special handling of components representing recurring items.

To target a property inside of a component, a property segment is added to the path. A property segment can include an optional match item. When present, this allows targeting of properties by value (matching or not matching a specific value), or which have a named property parameter present, or by property parameter value (matching or not matching a specific value).

To target a property parameter, a parameter segment is added to the property segment at the end of the path.

To target a single value in a multi-valued property, a value segment is added to the property segment at the end of the path.

To target a single value in a multi-valued property parameter, a value segment is added to the parameter segment at the end of the path.

Values in match items MUST use URL-style percent (%) encoding of the characters "/", "#", ";", "=", and "]". This allows a path to be quickly split into segments by breaking apart the text on the relevant delimiter characters.

The syntax for a path is defined by the following notation (note that some of the syntax elements defined here are not used by this specification, however, it is anticipated that this general path syntax will be useful for other specifications):

```
abs-path           = abs-comp-path [prop-all-path]
                    ; Absolute path for any iCalendar element

rel-full-path      = (comp-path [prop-all-path]) / prop-all-path
                    ; Relative path for any iCalendar element at any depth
                    ; within the enclosing component

rel-one-path       = comp-path / prop-all-path
                    ; Relative path for any iCalendar element immediately
                    ; within the enclosing component

abs-comp-path      = "/VCALENDAR" *comp-segment
                    ; Absolute path for components only

comp-path          = 1*comp-segment
                    ; Path for components only

prop-path          = prop-segment
                    ; Relative path for properties only

prop-param-path    = prop-segment [param-segment]
                    ; Relative path for property and parameter only
```

```
prop-all-path    = prop-segment [param-segment] [value-segment]
                    ; Relative path for any element of a property
```

```
comp-segment      = "/" name comp-match
comp-match        = [uid-match] [rid-match]
uid-match         = "[UID=" value-escaped "]"
rid-match         = "[RID=" ("M" / ridval) "]"
                    ; "M" matches "master" component
```

```
prop-segment      = prop-prefix [prop-match]
prop-prefix       = "#" name
prop-match        = "[" ( prop-equal / prop-not-equal /
                        param-match ) "]"
```

```
prop-equal        = "=" value-escaped
prop-not-equal    = "!" value-escaped
```

```
param-match       = "@" param-name [ ( param-equal /
                                       param-not-equal ) ]
param-equal       = "=" param-value-escaped
param-not-equal   = "!" param-value-escaped
```

```
param-segment     = ";" param-name
```

```
value-segment     = "=" (value / param-value)
```

```
value-escaped     = value
                    ; % encoding for "/", "#", ";", and "]"
```

```
param-value-escaped = param-value
                    ; % encoding for "/", "#", ";", and "]"
```

Some examples of "path" items follow.

Targeting components (path contains exactly one or more component segments):

- /VCALENDAR
Targets the "VCALENDAR" component in the iCalendar object.
- /VCALENDAR/VEVENT
Targets all "VEVENT" components in the "VCALENDAR" component in the iCalendar object.
- /VCALENDAR/VEVENT[UID=1234]
Targets any "VEVENT" components that have a "UID" property value exactly equal to "1234", in the "VCALENDAR" component in the iCalendar object.
- /VCALENDAR/VEVENT[UID=1234%2F4567] [RID=M]
Targets any "VEVENT" components that have a "UID" property value exactly equal to "1234/4567" and do not have a "RECURRENCE-ID" property, in the "VCALENDAR" component in the iCalendar object.
- `VCALENDAR/VEVENT[UID=1234][RID=20160902T223000Z]
Targets any "VEVENT" components that have a "UID" property value exactly equal to "1234" and have a "RECURRENCE-ID" property whose UTC value is "20160902T223000Z", in the "VCALENDAR" component in the iCalendar object.

Targeting properties (path contains exactly zero or more component segments, and one property segment):

- /VCALENDAR/VEVENT#STATUS
Targets all "STATUS" properties in all "VEVENT" components in the "VCALENDAR" component in the iCalendar object.
- /VCALENDAR/VEVENT[UID=1234]#STATUS

Targets all "STATUS" properties in any "VEVENT" components that have a "UID" property value exactly equal to "1234", in the "VCALENDAR" component in the iCalendar object.

- /VCALENDAR/VEVENT#ATTENDEE[=mailto:cyrus@example.com]
Targets any "ATTENDEE" properties that have the value "mailto:cyrus@example.com" in all "VEVENT" components, in the "VCALENDAR" component in the iCalendar object.
- /VCALENDAR/VEVENT#ATTENDEE[!mailto:cyrus@example.com]
Targets any "ATTENDEE" properties that do not have the value "mailto:cyrus@example.com" in all "VEVENT" components, in the "VCALENDAR" component in the iCalendar object.
- /VCALENDAR/VEVENT#ATTENDEE\<#xa0;<MEMBER>>
Targets any "ATTENDEE" properties that have a "MEMBER" property parameter present in all "VEVENT" components, in the "VCALENDAR" component in the iCalendar object
- /VCALENDAR/VEVENT#ATTENDEE<#xa0;<CN=Cyrus Daboo>>
Targets any "ATTENDEE" properties that have a "CN" property parameter with the value "Cyrus Daboo" present in all "VEVENT" components, in the "VCALENDAR" component in the iCalendar object.
- /VCALENDAR/VEVENT#ATTENDEE<#xa0;<CN!Cyrus Daboo>>
Targets any "ATTENDEE" properties that have a "CN" property parameter not equal to the value "Cyrus Daboo", or do not have a "CN" property parameter present in all "VEVENT" components, in the "VCALENDAR" component in the iCalendar object.
- #ATTENDEE[=mailto:cyrus@example.com]
A relative path that targets any "ATTENDEE" properties that have the value "mailto:cyrus@example.com" in all components the path is relative to.

Targeting property parameters (path contains exactly zero or more component segments, one property segment, and one parameter segment):

- /VCALENDAR/VEVENT#ATTENDEE;PARTSTAT
Targets the "PARTSTAT" parameter on all "ATTENDEE" properties in all "VEVENT" components in the "VCALENDAR" component in the iCalendar object.
- /VCALENDAR/VEVENT#ATTENDEE[=mailto:cyrus@example.com];PARTSTAT
Targets the "PARTSTAT" parameter on any "ATTENDEE" properties that have the value "mailto:cyrus@example.com" in all "VEVENT" components, in the "VCALENDAR" component in the iCalendar object.

Targeting property values (path contains exactly zero or more component segments, one property segment, and one value segment):

- /VCALENDAR/VEVENT#EXDATE=20160905
Targets all "EXDATE" property values with the value "20160905" in all "VEVENT" components in the "VCALENDAR" component in the iCalendar object.

Targeting property parameter values (path contains exactly zero or more component segments, one property segment, one parameter segment, and one value segment):

- /VCALENDAR/VEVENT#ATTENDEE;MEMBER=mailto:group@example.com
Targets all "MEMBER" property parameter values with the value "mailto:group@example.com" in all "ATTENDEE" properties in all "VEVENT" components in the "VCALENDAR" component in the iCalendar object.

6. Adding or Updating Components

Any iCalendar component defined in the "PATCH" component (referred to below as the "action component") is treated as either an addition to the target component, or as an update of an existing component in the target component. The following rules are used to process such components:

- If the action component contains a "UID" property and a "RECURRENCE-ID" property, then any components with the same values for both their "UID" and "RECURRENCE-ID" properties,

that are immediate sub-components of the target component, are removed from the target component, and the action component is added to the target component.

- If the action component contains a “UID” property and does not contain a “RECURRENCE-ID” property, then any components with the same value for their “UID” property, and containing no “RECURRENCE-ID” property, that are immediate sub-components of the target component, are removed from the target component, and the action component is added to the target component.
- If the action component does not contain a “UID” property, then all components with the same name that do not contain a “UID” property, that are immediate sub-components of the target component, are removed from the target component, and the action component is added to the target component.

7. Adding or Updating Properties

Any iCalendar property defined in the “PATCH” component (referred to below as the “action property”) is treated as either an addition to the target component, or as an update of an existing property in the target component. A “PATCH-ACTION” (Section 10.4) property parameter can be defined on action properties and is used to control how the action is processed. Any “PATCH-ACTION” property parameter MUST be removed from the action property when it is added to the target component. The following rules are used to process such properties:

- If the action property does not contain a “PATCH-ACTION” property parameter, or contains a “PATCH-ACTION” property parameter with the default value “BYNAME”, then all properties with the same name in the target component are removed, and the action property is added to the target component.
- If the action property contains a “PATCH-ACTION” property parameter with the value “CREATE”, then the action property is added to the target component.
- If the action property contains a “PATCH-ACTION” property parameter with the value “BYVALUE”, then all properties with the same name and same value in the target component are removed, and the action property is added to the target component.
- If the action property contains a “PATCH-ACTION” property parameter with the value starting with “BYPARAM”, then all properties with the same name and a property parameter that matches the one that is part of the “PATCH-ACTION” property value, in the target component are removed, and the action property is added to the target component.

The “PATCH-ACTION=BYNAME” operation is used for adding or updating “singleton” properties — properties that only appear once in a given iCalendar component (e.g., “DTSTART”, “DTEND”, “LOCATION”, etc).

The “PATCH-ACTION=CREATE” operation is used for adding “multi- occurring” properties — properties that can appear more than once in a given iCalendar component (e.g., “ATTENDEE”, “ATTACH”, “EXDATE”, etc).

The “PATCH-ACTION=BYVALUE” operation is used for updating a specific “multi-occurring” property that can be uniquely identified by its value (e.g., the “ATTENDEE” property can appear multiple times in a “VEVENT” component, but each property will have a unique value in that component). This operation cannot be used when the value of the property is being changed. Instead, the “PATCH-ACTION=BYPARAM” operation can be used to identify the target property.

The “PATCH-ACTION=BYPARAM” operation is used for updating a specific “multi-occurring” property that can be uniquely identified by a parameter value that is the same in the action and target properties.

There may be some situations where a multi-occurring property cannot be uniquely identified. In such cases, the solution to updating one or more of them is to use a “PATCH-ACTION=BYNAME” to replace all the existing properties with one new one, then use “PATCH-ACTION=CREATE” to add back others that are unchanged or also being updated. Whilst this is not ideal, it is anticipated that

these situations can be avoided by adding appropriate property parameters with unique values to help disambiguate the multi-occurring properties.

8. Deleting Components, Properties, or Property Parameters

The “PATCH-DELETE” property (defined in Section 10.3.2) is used to indicate deletion of iCalendar elements from the component identified by the “PATCH-TARGET” property in the same “PATCH” component as the “PATCH-DELETE” property. As such, the value of the “PATCH-DELETE” property is always a relative path (see Section 5) that refers to an element that is an immediate “child” of the target component.

The following operations are supported:

Delete components	The “PATCH-DELETE” path value targets components only. The matching components are removed from the “parent” target component.
Properties	The “PATCH-DELETE” path value targets properties only. The matching properties are removed from the “parent” target component.
Property parameters	The “PATCH-DELETE” path value targets property parameters on specific properties only. The matching property parameters are removed from the corresponding property.
Property values	The “PATCH-DELETE” path value targets a property value on specific multi-valued properties only. The matching property value is removed from the the corresponding property. If that results in a property with no value, that property is also removed from its “parent” target component.
Property parameter values	The “PATCH-DELETE” path value targets a property parameter value on a specific multi-valued property parameter on specific properties only. The matching property parameter value is removed from the corresponding property parameter. If that results in a property parameter with no value, that property parameter is also removed from from the corresponding property.

9. Adding or Updating Property Parameters

The “PATCH-PARAMETER” property (defined in Section 10.3.3) is used to indicate addition or update of property parameters and property parameter values to properties contained in the components identified by the “PATCH-TARGET” property in the same “PATCH” component as the “PATCH-PARAMETER” property. As such, the value of the “PATCH- PARAMETER” property is always a relative path (see Section 5) that refers to a property that is an immediate “child” of the target component.

The following operations are supported:

Add or update property parameters	The “PATCH-PARAMETER” path value targets a property only. Any property parameters defined on the “PATCH-PARAMETER” replace the matching parameters on the target property, or are added to the target property if no matching parameters exist.
Add a property parameter value	The “PATCH-PARAMETER” path value targets a multi-valued parameter only. The values in any property parameters defined on the “PATCH-PARAMETER” property are added to the corresponding property parameters of the target properties. If no corresponding property

parameter is defined on the target properties, then property parameters are created with the corresponding values.

10. iCalendar Extensions

This specification adds a new “VPATCH” calendar component to iCalendar. The “VPATCH” component is itself a container for a new “PATCH” sub-component.

11. VPATCH Component

Component Name	VPATCH
Purpose	Provide a grouping of “PATCH” sub-components that describe the patch actions to be performed.
Description	This component serves as a container for a series of “PATCH” sub-components, each specifying patch actions to be performed on a certain target element in an iCalendar object.
Format Definition	A “VPATCH” calendar component is defined by the following notation: <pre> vpatchc = "BEGIN" ":" "VPATCH" CRLF vpatchprop action "END" ":" "VPATCH" CRLF vpatchprop = *(; ; The following are REQUIRED, ; but MUST NOT occur more than once. ; dtstamp / uid / ; ; ; The following are OPTIONAL, ; but MUST NOT occur more than once. ; patch-version / patch-order / ; ; The following are OPTIONAL, ; and MAY occur more than once. ; other-prop ;) other-prop = (iana-prop / x-prop) action = *(patchc / iana-comp / x-comp) </pre>

11.1. PATCH Component

Component Name	PATCH
Purpose	Provide a set of components, properties, and property parameters to be added to, deleted from, or updated in the iCalendar object.

Description This component provides a grouping of patch actions to be performed within the scope of a set of components. If the "PATCH-TARGET" property matches one or more iCalendar components, then the target components are patched using the remaining properties and components. If there is no iCalendar component that matches the "PATCH-TARGET" property in the iCalendar object, the "PATCH" action MUST succeed without any changes being applied to the iCalendar object being patched.

Format Definition A "PATCH" calendar component is defined by the following notation:

```
patchc      = "BEGIN" ":" "PATCH" CRLF
              patchprop subcomp
              "END" ":" "PATCH" CRLF

patchprop   = *(
              ;
              ; The following is REQUIRED,
              ; but MUST NOT occur more than once.
              ;
              patchtarget /
              ;
              ; The following are OPTIONAL,
              ; and MAY occur more than once.
              ;
              patchdelete / patchparam / other-prop
              ;
              )

subcomp     = *(
              ;
              ; The following are OPTIONAL,
              ; and MAY occur more than once.
              ;
              eventc / todoc / journalc / freebusyc /
              timezonec / alarmc / standard / daylight /
              availabilityc / availablec /
              iana-comp / x-comp
              ;
              )
```

12. VPATCH Properties

The "VPATCH" properties are attributes that apply to the "VPATCH" component, as a whole. These properties do not appear within "VPATCH" sub-components. They SHOULD be specified after the "BEGIN:VPATCH" delimiter string and prior to any sub-component.

12.1. PATCH-VERSION Property

Property Name PATCH-VERSION

Purpose This property specifies the identifier corresponding to the highest version number of the "VPATCH" specification that is required in order to interpret the "VPATCH" component.

Value Type INTEGER

Property Parameters	IANA and nonstandard property parameters can be specified on this property.
Conformance	This property can be specified once in an "VPATCH" component. The default value is "1". This property MUST be specified if its value is greater than "1". Otherwise, this property is OPTIONAL.
Description	A value of "1" corresponds to this memo. See Section 3 for a description of how this property is used.
Format Definition	This property is defined by the following notation: <pre> patch-version = "PATCH-VERSION pverparam ":" pvalue CRLF pverparam = *(";" other-param) pvalue = "1" / pmaxver ; "1" signifies compliance with this memo pmaxver = <A IANA-registered VPATCH version> ; Maximum VPATCH version needed to process the VPATCH ; component.</pre>

EXAMPLE

The following is an example of this property:

```
PATCH-VERSION:1
```

12.2. PATCH-ORDER Property

Property Name	PATCH-ORDER
Purpose	This property specifies the ordering of the associated "VPATCH" component.
Value Type	INTEGER
Property Parameters	IANA and nonstandard property parameters can be specified on this property.
Conformance	This property can be specified once in a "VPATCH" component.
Description	This property is OPTIONAL and is used to indicate the relative ordering of the associated "VPATCH" component amongst its siblings. See Section 3 for a description of how this property is used.
Format Definition	This property is defined by the following notation: <pre> patch-order = "PATCH-ORDER porderparam ":" integer CRLF porderparam = *(";" other-param)</pre>

EXAMPLE

The following is an example of this property:

PATCH-ORDER:1

13. PATCH Component Properties

The following properties can appear within PATCH components.

13.1. PATCH-TARGET Property

Property Name	PATCH-TARGET
Purpose	This property specifies a path targeting one or more components within an iCalendar object.
Value Type	TEXT
Property Parameters	IANA and nonstandard property parameters can be specified on this property.
Conformance	This property MUST be specified within any "PATCH" sub- component.
Description	This property is used to match iCalendar components that the patch operations will be applied to. The path value is always an absolute path, and interpreted as described in Section 5.
Format Definition	This property is defined by the following notation: <pre> patchtarget = "PATCH-TARGET ptargetparam ":" ptargetpath CRLF ptargetparam = *(";" other-param) ptargetpath = abs-comp-path / comp-path ; This specification only defines how abs- comp-path ; is used. Use of the comp-path element will be ; defined by other specifications wishing to make use ; of "relative" patches. </pre>

Example: The following is an example of this property:

```
PATCH-TARGET: /VCALENDAR/VEVENT[UID=1234]
```

13.2. PATCH-DELETE Property

Property Name	PATCH-DELETE
Purpose	This property specifies a path (relative to "PATCH-TARGET") targeting one or more components, properties, or parameters to be removed from the target components identified by "PATCH-TARGET".
Value Type	TEXT
Property Parameters	IANA and nonstandard property parameters can be specified on this property.
Conformance	This property can be specified within a "PATCH" sub- component.

Description This property is used to match iCalendar elements that will be deleted. The path value is always a relative path for only immediate components and properties within the target component, and interpreted as described in Section 8.

Format Definition This property is defined by the following notation:

```
patchdelete = "PATCH-DELETE pdeleteparam ":" pdeletepath
CRLF
```

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

```
pdeletepath = rel-one-path
; PATCH-DELETE path is relative to PATCH-
TARGET path
```

Example The following are examples of this property:

```
PATCH-DELETE:/VEVENT[UID=1234] PATCH-DELETE:#ATTENDEE[=mailto:cyrus@example.com]
```

13.3. PATCH-PARAMETER Property

Property Name PATCH-PARAMETER

Purpose This property specifies a set of parameters to be set on the target property.

Value Type TEXT

Property Parameters IANA and nonstandard property parameters can be specified on this property.

Conformance This property can be specified within a "PATCH" sub- component.

Description This property specifies parameters to be set on the target property. The path value is always a relative path to a property within the target component, and interpreted as described in Section 9.

Format Definition This property is defined by the following notation:

```
patchparam = "PATCH-PARAMETER pparamparam ":" pparampath
CRLF
```

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

```
pparampath = prop-param-path
```

EXAMPLE

The following are examples of this property:

```
PATCH-PARAMETER;PARTSTAT=NEEDS-ACTION:
#ATTENDEE[=mailto:cyrus@example.com]
PATCH-PARAMETER;PARTSTAT=NEEDS-ACTION:#ATTENDEE<&#xa0;<CN=Cyrus Daboo>>
PATCH-PARAMETER;MEMBER=mailto:newgroup@example.com:#ATTENDEE;MEMBER
```

13.4. PATCH-ACTION Property Parameter

Parameter Name PATCH-ACTION

Purpose	To specify whether the property should be added or replaced.
Description	This parameter can be specified on properties contained in a "PATCH" component and MUST NOT be specified on properties outside of a "PATCH" component. This parameter specifies whether the property should be added to the target component or should replace existing properties in the target component. In the latter case, the parameter also specifies how to match existing properties. The processing of this property parameter is described in Section 7.
Format Definition	This parameter is defined by the following notation: <pre> pactionparam = "PATCH-ACTION" "=" pactioncreate / pactionbyname / pactionbyvalue / pactionbyparam / iana-token / ; IANA registered value x-name ; Experimental value pactioncreate = "CREATE" ; Always add property to target component. pactionbyname = "BYNAME" ; Always remove properties with the same name ; from the target component, ; then add this property to the target component. ; This value is the default and MAY be omitted. pactionbyvalue = "BYVALUE" ; Always remove properties with the same name ; and value from the target component, ; then add this property to the target component. pactionbyparam = DQUOTE "BYPARAM" param-match DQUOTE ; Always remove properties with the same name ; and parameter name/value from the target ; component, then add this property to the target ; component.</pre>

EXAMPLE

The following are examples of this property parameter:

```

ATTENDEE;PATCH-ACTION=BYVALUE;PARTSTAT=NEEDS-ACTION:
mailto:cyrus@example.com
DESCRIPTION;PATCH-ACTION="BYPARAM@LANGUAGE=en_GB";LANGUAGE=en_US:
Meeting to discuss VPATCH
```

14. Additional Considerations

14.1. Handling Default Properties and Parameters

iCalendar properties and property parameters can have default values, which allows those items to be omitted from the iCalendar data, but with the default value assumed. A patch operation might add properties or property parameters with default values. A patch processing engine MAY choose

to remove properties or property parameters with default values from the patched iCalendar object.

14.2. Handling Recurrences

Recurring events (or other types of component) in iCalendar are defined by the presence of "RRULE", "RDATE", and "EXDATE" properties in a "master" iCalendar component. Those rules produce a set of "generated" instances. In some cases specific "generated" instances are changed, resulting in the presence of "overridden" components, which are identified by having the same "UID" property value as the "master" component, and a "RECURRENCE-ID" property whose value matches the start time of the corresponding "generated" instance (which can be different from the actual start time of the overridden instance).

When a set of master and overridden recurring components exist in the iCalendar object being patched, each can be uniquely targeted by using the "RID=" match item in the component segment of the path value of a "PATCH-TARGET" or "PATCH-DELETE" property. To target the master component, a "RID=M" match item is used. To target an overridden component, the "RID=" value is set to the value of the "RECURRENCE-ID" property in the overridden component.

Patch commands can also be used to implicitly create overridden components in the iCalendar object being patched by specifying a path with a "RID=" match item, using what would be the overridden component's "RECURRENCE-ID" value if it existed as a separate component. This is useful when an overridden component needs to be added, but the changes to it are small (e.g., an instance where only the summary of the event is different).

If the value of a "RID=" match item in a path does not correspond to an existing instance (either because its value does not match a "generated" instance, or its value matches an "EXDATE" in the "master" component), then the patch operation MUST fail.

For example, consider the following daily recurring event:

```
BEGIN:VCALENDAR
PRODID:test
VERSION:2.0
BEGIN:VEVENT
UID:1234
DTSTART:20160902T120000Z
DURATION:PT1H
SUMMARY:Master component
RRULE:FREQ=DAILY
END:VEVENT
END:VCALENDAR
```

The following patch command could be used to update the "SUMMARY" property value of the second instance of the recurring event:

```
BEGIN:VPATCH
UID:abcd
DTSTAMP:20160901T000000Z
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR/VEVENT[UID=1234][RID=20160903T120000Z]
SUMMARY:Override second instance
END:PATCH
END:VPATCH
```

which results in the following updated iCalendar component:

```
BEGIN:VCALENDAR
PRODID:test
VERSION:2.0
BEGIN:VEVENT
```

```

UID:1234
DTSTART:20160902T120000Z
DURATION:PT1H
SUMMARY:Master component
RRULE:FREQ=DAILY
END:VEVENT
BEGIN:VEVENT
UID:1234
RECURRENCE-ID=20160903T120000Z
DTSTART:20160903T120000Z
DURATION:PT1H
SUMMARY:Override second instance
END:VEVENT
END:VCALENDAR

```

A similar result could have been achieved by using a path targeting the “VCALENDAR” component, and the entire “overridden” component supplied as the data. However, the implicit override behaviour allows for a more compact representation of this type of change.

There is no equivalent behavior when it comes to removing “overridden” components from an iCalendar object to cancel the instance. In that case, two “PATCH” components are required: one to delete the “overridden” component, and one to create an “EXDATE” property value in the master component to cover the cancellation. So, continuing from the example data immediately above, the following patch commands would cancel the instance that was previously overridden:

```

BEGIN:VPATCH
UID:abcd
DTSTAMP:20160901T000000Z
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR
PATCH-DELETE:/VEVENT[UID=1234][RID=20160903T120000Z]
END:PATCH
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR/VEVENT[UID=1234][RID=M]
EXDATE;PATCH-ACTION=CREATE:20160903T120000Z
END:PATCH
END:VPATCH

```

which results in the following updated iCalendar component:

```

BEGIN:VCALENDAR
PRODID:test
VERSION:2.0
BEGIN:VEVENT
UID:1234
DTSTART:20160902T120000Z
DURATION:PT1H
SUMMARY:Master component
RRULE:FREQ=DAILY
EXDATE:20160903T120000Z
END:VEVENT
END:VCALENDAR

```

14.3. Folded lines

iCalendar data can contain “folded” lines (as described in Section 3.1 of [IETF RFC 5545](https://tools.ietf.org/html/rfc5545)). The patch operations described in this specification are a “semantic” rather than “syntactic” update to the data. i.e., they apply to the underlying object model as opposed to the “raw” iCalendar text data. As such, folded lines in the iCalendar data targeted by the patch commands are not significant.

Any iCalendar data supplied as data items in a patch command MAY contain folded lines.

14.4. Encoding

Text values in iCalendar use a backslash escape mechanism for certain characters (as described in Section 3.3.11 [IETF RFC 5545](#)). Patch operations apply to the escaped form of the iCalendar data. For example, to delete a “DESCRIPTION” property that contains an encoded line feed character:

```
DESCRIPTION:Line one\nLine two
```

the following PATCH-DELETE property would be used:

```
PATCH-DELETE:#DESCRIPTION[=Line one\nLine two]
```

Similarly, to update the “DESCRIPTION” property, the following patch command could be used:

```
BEGIN:VPATCH
UID:abcd
DTSTAMP:20160901T000000Z
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR/VEVENT
DESCRIPTION:Line one\nLine two\nLine three
END:PATCH
END:VPATCH
```

14.5. Generation

This specification does not define how patch data is generated, as that is likely to be highly dependent on the nature of the implementation. However, it is recommended that patch generators use sets of commands that keep the overall patch data as compact as possible, since one of the goals of this specification is to reduce the size of data needed to do updates. One example is the choice of whether to update an entire property, or just property parameters, when changes are made to just property parameters. In some cases, the data in a property parameter can be large, so repeating that in a full property update may result in larger data than simple using the “PATCH-PARAMETER” property to do an update. On the other hand, if lots of property parameters are being updated or removed, it can be more efficient to update the entire property rather than using lots of “PATCH-PARAMETER” and “PATCH-DELETE” properties.

15. Use with iTIP

iTIP [IETF RFC 5546](#) defines how iCalendar data can be sent between calendar user agents to schedule calendar components between calendar users. This specification does not define how iCalendar patch documents can be used with iTIP.

16. Use with CalDAV and HTTP

The CalDAV [IETF RFC 4791](#) calendar access protocol allows clients and servers to exchange iCalendar data. iCalendar data is typically stored in calendar object resources on a CalDAV server. A CalDAV client typically updates the calendar object resource data via an HTTP PUT request, which requires sending the entire iCalendar object in the HTTP request body.

A server can also support the HTTP PATCH method [IETF RFC 5789](#) which allows a patch document to be specified in the request body, and for that patch document to be applied to the resource targeted by the HTTP request. In this case, the server would advertise the “text/calendar” media type in an “Accept-Patch” header field as described in Section 3.1 of [IETF RFC 5789](#). Note that the requirements for parameters on this media type when advertised in “Accept-Patch” are as follows:

- MUST include a “component” parameter with a value of “VPATCH”

- MUST include an “optinfo” parameter with a value of “PATCH-VERSION:{n}”, where “{n}” is the maximum patch version supported by the server
- MAY include a “charset” parameter as appropriate

EXAMPLE

```
Accept-Patch: text/calendar; component=VPATCH;
optinfo="PATCH-VERSION:1"; charset=utf-8
```

The PATCH-TARGET property defined by this specification does not allow targeting the entire iCalendar object, and hence an HTTP PATCH request cannot be used to create a new resource (a normal HTTP PUT request is used instead).

17. Security Considerations

Patch processing engines MUST ensure that the result of applying a patch is a valid iCalendar object in the context of the application using the calendar data. At the very least, the resulting iCalendar object MUST comply with the requirements of [IETF RFC 5545](#).

Security considerations described in [IETF RFC 5545](#), [IETF RFC 5789](#), and [IETF RFC 4791](#) MUST be adhered to.

18. Privacy considerations

Privacy considerations described in [IETF RFC 5545](#), [IETF RFC 5789](#), and [IETF RFC 4791](#) MUST be adhered to.

19. IANA Considerations

19.1. Component Registrations

This document defines the following new iCalendar components to be added to the registry defined in Section 8.3.1 of [IETF RFC 5545](#):

Table 1

Component	Status	Reference
VPATCH	Current	RFCXXXX, Section 10.1
PATCH	Current	RFCXXXX, Section 10.1.1

19.2. Property Registrations

This document defines the following new iCalendar properties to be added to the registry defined in Section 8.3.2 of [IETF RFC 5545](#):

Table 2

Property	Status	Reference
PATCH-VERSION	Current	RFCXXXX, Section 10.2.1
PATCH-ORDER	Current	RFCXXXX, Section 10.2.2
PATCH-TARGET	Current	RFCXXXX, Section 10.3.1
PATCH-DELETE	Current	RFCXXXX, Section 10.3.2
PATCH-PARAMETER	Current	RFCXXXX, Section 10.3.3

19.3. Parameter Registrations

This document defines the following new iCalendar parameters to be added to the registry defined in Section 8.3.3 of [IETF RFC 5545](#):

Table 3

Property	Status	Reference
PATCH-ACTION	Current	RFCXXXX, Section 10.4

19.4. Property and Parameter Value Registries

Two new IANA registries for iCalendar elements have been added. Additional codes MAY be used, provided the process described in Section 8.2.1 of [IETF RFC 5545](#) is used to register them, using the template in Section 8.2.6 of [IETF RFC 5545](#).

19.4.1. Patch Version Registry

The following table has been used to initialize the Patch Version Registry:

Table 4

Patch Version	Status	Reference
1	Current	RFCXXXX

19.4.2. Patch Action Registry

The following table has been used to initialize the Patch Action Registry:

Table 5

Patch Action	Status	Reference
CREATE	Current	RFCXXXX, Section 10.4
BYNAME	Current	RFCXXXX, Section 10.4
BYVALUE	Current	RFCXXXX, Section 10.4
BYPARAM	Current	RFCXXXX, Section 10.4

20. VPATCH Examples

Examples of single command patch documents for common iCalendar data operations.

20.1. Add a new component

Creates a new "VEVENT" component.

```
BEGIN:VCALENDAR
PRODID:Example
VERSION:2.0
BEGIN:VPATCH
UID:abcd
DTSTAMP:20160901T000000Z
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR
BEGIN:VEVENT
UID:1234
DTSTART:20160902T103000Z
DURATION:PT1H
SUMMARY:Test event
```

```
END:VEVENT
END:PATCH
END:VPATCH
END:VCALENDAR
```

20.2. Add a new VALARM component

Creates a new “VALARM” component in the “VEVENT” component with the “UID” property value “1234”.

```
BEGIN:VCALENDAR
PRODID:Example
VERSION:2.0
BEGIN:VPATCH
UID:abcd
DTSTAMP:20160901T000000Z
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR/VEVENT[UID=1234]
BEGIN:VALARM
UID:4567
ACTION:DISPLAY
TRIGGER:-PT30M
DESCRIPTION:Time to leave
END:VALARM
END:PATCH
END:VPATCH
END:VCALENDAR
```

20.3. Replace a component

Replace the “VEVENT” component with the “UID” property value “1234” with a new component.

```
BEGIN:VCALENDAR
PRODID:Example
VERSION:2.0
BEGIN:VPATCH
UID:abcd
DTSTAMP:20160901T000000Z
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR/VEVENT[UID=1234]
BEGIN:VEVENT
UID:1234
DTSTART:20160903T123000Z
DURATION:PT2H
SUMMARY:Changed event
END:VEVENT
END:PATCH
END:VPATCH
END:VCALENDAR
```

20.4. Remove a component

Remove the “VEVENT” component with the “UID” property value “1234”.

```
BEGIN:VCALENDAR
PRODID:Example
VERSION:2.0
BEGIN:VPATCH
UID:abcd
```

CC/WD 51012:2016

```
DTSTAMP:20160901T000000Z
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR
PATCH-DELETE:/VEVENT[UID=1234]
END:PATCH
END:VPATCH
END:VCALENDAR
```

20.5. Add properties to a component

Add “STATUS” and “COMPLETED” properties to the “VTODO” component with the “UID” property value “4321”.

```
BEGIN:VCALENDAR
PROID:Example
VERSION:2.0
BEGIN:VPATCH
UID:abcd
DTSTAMP:20160901T000000Z
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR/VTODO[UID=4321]
STATUS;PATCH-ACTION=CREATE:COMPLETED
COMPLETED;PATCH-ACTION=CREATE:20160902T224515Z
END:PATCH
END:VPATCH
END:VCALENDAR
```

20.6. Update properties in a component

Update the “SUMMARY” and “LOCATION” properties in the “VEVENT” component with the “UID” property value “1234”.

```
BEGIN:VCALENDAR
PROID:Example
VERSION:2.0
BEGIN:VPATCH
UID:abcd
DTSTAMP:20160901T000000Z
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR/VEVENT[UID=1234]
SUMMARY:Title was changed
LOCATION:New place
END:PATCH
END:VPATCH
END:VCALENDAR
```

20.7. Update a targeted property in a component

Update the “ATTENDEE” property with value “mailto:cyrus@example.com” in the “VEVENT” component with the “UID” property value “1234”.

```
BEGIN:VCALENDAR
PROID:Example
VERSION:2.0
BEGIN:VPATCH
UID:abcd
DTSTAMP:20160901T000000Z
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR/VEVENT[UID=1234]
```

```

ATTENDEE;PATCH-ACTION=BYVALUE;PARTSTAT=ACCEPTED:
mailto:cyrus@example.com
END:PATCH
END:VPATCH
END:VCALENDAR

```

20.8. Remove a property from a component

Remove the “URL” property from the “VEVENT” component with the “UID” property value “1234”.

```

BEGIN:VCALENDAR
PRODID:Example
VERSION:2.0
BEGIN:VPATCH
UID:abcd
DTSTAMP:20160901T000000Z
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR/VEVENT[UID=1234]
PATCH-DELETE:#URL
END:PATCH
END:VPATCH
END:VCALENDAR

```

20.9. Remove a property with a specific value from a component

Remove the “ATTENDEE” property with the value “mailto:cyrus@example.com” in the “VEVENT” component with the “UID” property value “1234”.

```

BEGIN:VCALENDAR
PRODID:Example
VERSION:2.0
BEGIN:VPATCH
UID:abcd
DTSTAMP:20160901T000000Z
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR/VEVENT[UID=1234]
PATCH-DELETE:#ATTENDEE[=mailto:cyrus@example.com]
END:PATCH
END:VPATCH
END:VCALENDAR

```

20.10. Change a parameter on a property with a specific value from a component

Change or add the “PARTSTAT” parameter on the “ATTENDEE” property with the value “mailto:cyrus@example.com” in the “VEVENT” component with the “UID” property value “1234”.

```

BEGIN:VCALENDAR
PRODID:Example
VERSION:2.0
BEGIN:VPATCH
UID:abcd
DTSTAMP:20160901T000000Z
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR/VEVENT[UID=1234]
PATCH-PARAMETER;PARTSTAT=ACCEPTED:
=ATTENDEE[=mailto:cyrus@example.com]
END:PATCH
END:VPATCH

```

END:VCALENDAR

20.11. Remove a parameter on a property with a specific value from a component

Remove the "PARTSTAT" parameter from the "ATTENDEE" property with the value "mailto:cyrus@example.com" in the "VEVENT" component with the "UID" property value "1234".

```
BEGIN:VCALENDAR
PRODID:Example
VERSION:2.0
BEGIN:VPATCH
UID:abcd
DTSTAMP:20160901T000000Z
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR/VEVENT[UID=1234]
PATCH-DELETE:#ATTENDEE[=mailto:cyrus@example.com];PARTSTAT
END:PATCH
END:VPATCH
END:VCALENDAR
```

21. Remove a value from a multi-valued parameter on a property with a specific value from a component

Remove the "mailto:calext@example.com" value from the "MEMBER" parameter on the "ATTENDEE" property with the value "mailto:cyrus@example.com" in the "VEVENT" component with the "UID" property value "1234".

```
BEGIN:VCALENDAR
PRODID:Example
VERSION:2.0
BEGIN:VPATCH
UID:abcd
DTSTAMP:20160901T000000Z
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR/VEVENT[UID=1234]
PATCH-DELETE:#ATTENDEE[=mailto:cyrus@example.com]
;MEMBER=mailto:calext@example.com
END:PATCH
END:VPATCH
END:VCALENDAR
```

21.1. Remove a value from a multi-valued property from a component

Remove the value "20160903T120000Z" from the "EXDATE" property in the "VEVENT" component with the "UID" property value "1234".

```
BEGIN:VCALENDAR
PRODID:Example
VERSION:2.0
BEGIN:VPATCH
UID:abcd
DTSTAMP:20160901T000000Z
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR/VEVENT[UID=1234]
PATCH-DELETE:#EXDATE=20160903T120000Z
END:PATCH
END:VPATCH
```

END:VCALENDAR

21.2. Attendee updating their participation status

When an attendee updates their participation status in an event, they will typically: update the "PARTSTAT" parameter on their "ATTENDEE" property, remove the "RSVP" parameter on their "ATTENDEE" property, update the "TRANSP" property in the "VEVENT" component. This set of changes is shown below in a single "PATCH" component, with the attendee having the calendar user address "mailto:cyrus@example.com". The patch targets all "VEVENT" components in the iCalendar object being changed.

```
BEGIN:VCALENDAR
PRODID:Example
VERSION:2.0
BEGIN:VPATCH
UID:abcd
DTSTAMP:20160901T000000Z
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR/VEVENT
PATCH-DELETE:#ATTENDEE[=mailto:cyrus@example.com];RSVP
PATCH-PARAMETER;PARTSTAT=ACCEPTED:
=ATTENDEE[=mailto:cyrus@example.com]
TRANSP:OPAQUE
END:PATCH
END:VPATCH
END:VCALENDAR
```

21.3. Recurring event adding one override

A daily recurring "VEVENT" component with the "SUMMARY" property being overridden for the second instance.

iCalendar object before the patch:

```
BEGIN:VCALENDAR
PRODID:Example
VERSION:2.0
BEGIN:VEVENT
UID:1234
DTSTART:20160905
DURATION:PT1H
SUMMARY:Test event
RRULE:FREQ=DAILY
END:VEVENT
END:VCALENDAR
```

Patch:

```
BEGIN:VCALENDAR
PRODID:Example
VERSION:2.0
BEGIN:VPATCH
UID:abcd
DTSTAMP:20160901T000000Z
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR/VEVENT[RID=20160906]
SUMMARY:Test event - modified
END:PATCH
END:VPATCH
```

END:VCALENDAR

iCalendar object after the patch:

```
BEGIN:VCALENDAR
PRODID:Example
VERSION:2.0
BEGIN:VEVENT
UID:1234
DTSTART:20160905
DURATION:PT1H
SUMMARY:Test event
RRULE:FREQ=DAILY
END:VEVENT
BEGIN:VEVENT
UID:1234
RECURRENCE-ID:20160906
DTSTART:20160905
DURATION:PT1H
SUMMARY:Test event - modified
END:VEVENT
END:VCALENDAR
```

21.4. Removal of an overridden instance

A daily recurring “VEVENT” component has one existing instance override removed with an “EXDATE” added for it.

iCalendar object before the patch:

```
BEGIN:VCALENDAR
PRODID:Example
VERSION:2.0
BEGIN:VEVENT
UID:1234
DTSTART:20160905
DURATION:PT1H
SUMMARY:Test event
RRULE:FREQ=DAILY
END:VEVENT
BEGIN:VEVENT
UID:1234
RECURRENCE-ID:20160906
DTSTART:20160905
DURATION:PT1H
SUMMARY:Test event - modified
END:VEVENT
END:VCALENDAR
```

Patch:

```
BEGIN:VCALENDAR
PRODID:Example
VERSION:2.0
BEGIN:VPATCH
UID:abcd
DTSTAMP:20160901T000000Z
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR
PATCH-DELETE:/VEVENT[RID=20160906]
END:PATCH
```

```
BEGIN:PATCH
PATCH-TARGET:/VCALENDAR/VEVENT[RID=M]
EXDATE;PATCH-ACTION=CREATE:20160906
END:PATCH
END:VPATCH
END:VCALENDAR
```

iCalendar object after the patch:

```
BEGIN:VCALENDAR
PRODID:Example
VERSION:2.0
BEGIN:VEVENT
UID:1234
DTSTART:20160905
DURATION:PT1H
SUMMARY:Test event
RRULE:FREQ=DAILY
EXDATE:20160906
END:VEVENT
END:VCALENDAR
```

22. Acknowledgements

Thanks to the following for feedback: Michael Douglass

This specification originated from work at the Calendaring and Scheduling Consortium [CalConnect](#), which has helped with the development and testing of implementations.

Appendix A (normative)

Change History (To be removed by RFC Editor before publication)

Changes in draft-daboo-icalendar-vpatch-00:

- Allow PATCH-TARGET to use comp-path relative paths.
- Fix uid-match to use escaped values.

Bibliography

- [1] IETF RFC 2119, BRADNER, S. *Key words for use in RFCs to Indicate Requirement Levels*. 1997. RFC Publisher. <https://www.rfc-editor.org/info/rfc2119>.
- [2] IETF RFC 5789, DUSSEAULT, L. and J. SNELL. *PATCH Method for HTTP*. 2010. RFC Publisher. <https://www.rfc-editor.org/info/rfc5789>.
- [3] CalConnect, *The Calendaring and Scheduling Consortium*