Network Working Group
Internet-Draft
Updates: 5545 (if approved)
Intended status: Standards Track
Expires: May 2, 2020

M. Douglass
Spherical Cow Group

October 30, 2019

Support for Series in iCalendar
draft-ietf-calext-icalendar-series-00

Abstract

This specification updates [RFC5545] by defining a new repeating set of events known as a series. This differs from recurrences in that each instance is a separate entity with a parent relationship to a specified template entity.

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 https://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 May 2, 2020.

Table of Contents

1. Introduction

Since iCalendar was first defined there has been only one standard way to express a repeating set of events - the recurrence. This defined a master event, a set of rules for computing the instances and a way of overriding certain instances.

This approach works well enough in certain situations but has many problems which need to be addressed.

This specification introduces a new approach to repeating patterns of entities which avoids some of the problems.

2. Overrides and iCalendar recurrences

The recurrence rules specify how instances are to be computed. These rules provide a set of keys - the RECURRENCE-IDs - and an instance can be created with the calculated start date/time and a copy of the duration (or calculated end date/time).

The specification allows for overrides. These are handled by supplying a complete replacement for the instance with a RECURRENCE-ID property matching that of the instance being overridden. This may change any of the properties (except the UID) - including start, end or duration.

If a long lived recurrence is heavily overridden it becomes very cumbersome. The master plus overrides is considered a single resource in most circumstances (however iTip allows the delivery of a single instance in certain situations).

Simple meetings can become heavily modifed recurrences through adding the weeks agenda to the description, changing of attendees etc.

There are approaches being considered to mitigate some of these issues which mostly involve only storing differences. Depending on the actual changes this may be more or less succesful. However recurrences are still awkward to deal with.

2.1. Changing the master start or the recurrence rules

This can lead to some very difficult problems to resolve. In the case of a heavily modified meeting it may be difficult to impossible to determine which override applies to the newly modified event.

For example, a weekly book-reading is moved from Monday to Friday. There are weeks of scheduled events in the future. Do we move them all forward to the next instance or skip one and move them back? If it becomes bi-weekly rather than weekly do we drop every other or just space them out more?

To be sure - some of these problems are not totally resolved by a series approach but they become more tractable. An approach on how to mitigate these problems is defined later on in this specification.

2.2. Splitting recurrences

The [RFC5545] THISANDFUTURE range is poorly supported. Splitting is what a number of implementations use to avoid changing overrrides in the past.

The recurring event is split into 2, one being the truncated original the other being a new recurring event starting at the time of the THISANDFUTURE override.

There is left the problem of relating the two, this can be accomplished by use of the RELATED-TO property but that is not standardized.

3. Series

A series is a, generally regularly, repeating sets of events or tasks each instance of which is usually, but not always, different in some respect. Examples may be a library running an after-school reading program which usually, takes place at the same time each week but always differs in the book or author being studied.

In recurrences an instances is a calculated 'virtual' object, unless overridden. It has the same UID as the master and a RECURRENCE-ID which is always one of the calculated set.

In a series, a specified number of instances are created ahead of time each with their own unique UID. They are all related to the master using a SERIES-MASTER relation type defined in this specification. Each instance acts as an individual component as far as retrieval and searching is concerned.

Each instance and master is identified as a member of the full series by the SERIES-UID property. The value of this property is the same in all members of the series even when splits have occurred.

As instances are created a LAST-SERIES-ID property is added or updated in the master to indicate which instance was last created. When there are SXDATE properties this property value may represent an instance which cannot be created. It merely represents the latest calculated date.

This property allows generated instances to be deleted without the addition of SXDATE properties to the master. The SXDATE only indicates future instances which MUST NOT be created.

As time goes on more instances are created either by the server or by a client when it inspects the current state of the series. The number of instances may be based on time or a count.

For example, an organization may allow rooms to be booked only 4 weeks ahead. Thus a series may be set up which has that 4 week set of events in the future. Each will have the room as an attendee ensuring that at least the room is booked at the regular time.

This does not prevent a client or server from creating dummy events out into the future as a guide for people managing their calendars. The application or server merely has to ensure that those dummy events are marked as such and are read-only.

To facilitate this there is a new SITEM-STATUS property which may be used to mark such an instance.

3.1. Modifying series patterns and splitting

If it becomes necessary to modify the series rules or the master start then the series is always split at the point of the modification.

When a series is split the previous master is modifed to truncate the current series at the last generated instance and a parameter SPLIT=YES is added to the series rule to indicate that this master is now split.

The split may result in a number of instances related to the old series but overlapping the new. It is up to the implementation to decide what should be done with these but this usually requires a degree of interaction with a human (or very intelligent robot). The application may offer to copy them into the corresponding new instances - if these can be easily determined, offer to delete all of them or let the user manually copy information and delete.

The new series master is related to the old master by the new series master having a RELATED-TO property with RELTYPE=SERIES-MASTER pointing at the previous master. In that way a backwards chain of series masters may be created

3.2. The series master

A series master is identified in much the same way as a recurrence master. It will contain an SRULE and 0 or more SDATE properties or 1 or more SDATE properties. Additionally it may contain 0 or more SXDATE properties to exclude instances.

As noted above, if the series was split it may contain a RELATED-TO property with RELTYPE=SERIES-MASTER and a value of the previous series master.

The master will also contain a LAST-SERIES-ID if any instances have been calculated and perhaps generated.

It is important to note that the series master is NOT a member of the series. This makes it easier for services to filter out series masters.

3.3. The series instances

A series instance is identified by having a SERIES-ID property which is calculated in the same manner as a RECURRENCE-ID. It MUST also contain a RELATED-TO property with RELTYPE=SERIES-MASTER and a value being the UID of the series master.

As noted above, if the series was split it may contain a RELATED-TO property with RELTYPE=SERIES-MASTER and a value being the UID of the previous series master.

4. Redefined Relation Type Value

Relationship parameter type values are defined in section 3.2.15. of [RFC5545]. This specification augments that parameter to include the new relationship values SERIES-MASTER

Format Definition:

This property parameter is respecified as follows:

```
reltypeparam   = "RELTYPE" "="
                  ("PARENT"    ; Parent relationship - Default
                 / "CHILD"     ; Child relationship
                 / "SIBLING"   ; Sibling relationship
                 / "DEPENDS-ON" ; refers to previous task
                 / "REFID"     ; Relationship based on REFID
                 / "STRUCTURED-CATEGORY"
                        ; Relationship based on STRUCTURED-CATEGORY
                 / "FINISHTOSTART" ;  Temporal relationship
                 / "FINISHTOFINISH" ;  Temporal relationship
                 / "STARTTOFINISH" ;  Temporal relationship
                 / "STARTTOSTART" ;  Temporal relationship
                 / "SERIES-MASTER" ;  link to the master component
                 / iana-token  ; Some other IANA-registered
                               ; iCalendar relationship type
                 / x-name)     ; A non-standard, experimental
                               ; relationship type
```

Description: This parameter can be specified on a property that references another related calendar component. The parameter may specify the hierarchical relationship type of the calendar component referenced by the property when the value is PARENT, CHILD or SIBLING. If this parameter is not specified on an allowable property, the default relationship type is PARENT. Applications MUST treat x-name and iana-token values they don't recognize the same way as they would the PARENT value.

This parameter defines the temporal relationship when the value is one of the project management standard relationships FINISHTOSTART, FINISHTOFINISH, STARTTOFINISH or STARTTOSTART. This property will be present in the predecessor entity and will refer to the successor entity. The GAP parameter specifies the lead or lag time between the predecessor and the successor. In the description of each temporal relationship below we refer to Task-A which contains and controls the relationship and Task-B the target of the relationship.

RELTYPE=PARENT: See [RFC5545] section 3.2.15.

RELTYPE=CHILD: See [RFC5545] section 3.2.15.

RELTYPE=SIBLING: See [RFC5545] section 3.2.15.

RELTYPE=DEPENDS-ON: Indicates that the current calendar component depends on the referenced calendar component in some manner. For

example a task may be blocked waiting on the other, referenced, task.

RELTYPE=REFID: Establishes a reference from the current component to components with a REFID property which matches the value given in the associated RELATED-TO property.

RELTYPE=SERIES-MASTER: Indicates that the current calendar component is bsed on the referenced calendar component. The value is a UID.

RELTYPE=STRUCTURED-CATEGORY: Establishes a reference from the current component to components with a STRUCTURED-CATEGORY property which matches the value given in the associated RELATED-TO property.

RELTYPE=FINISHTOSTART: Task-B cannot start until Task-A finishes. For example, when sanding is complete, painting can begin.

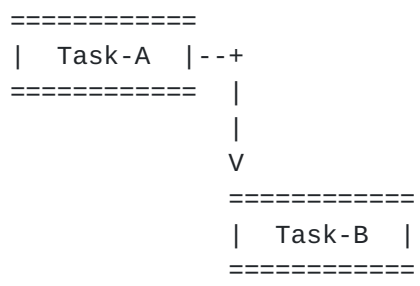

Figure 1: Finish to start relationship

RELTYPE=FINISHTOFINISH: Task-B cannot finish before Task-A is finished, that is the end of Task-A defines the end of Task-B. For example, we start the potatoes, then the meat then the peas but they should all be cooked at the same time.

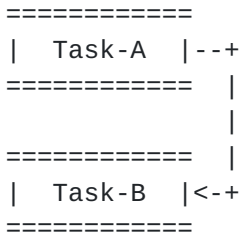

Figure 2: Finish to finish relationship

RELTYPE=STARTTOFINISH: The start of Task-A (which occurs after Task-B) controls the finish of Task-B. For example, ticket sales (Task-B) end when the game starts (Task-A).

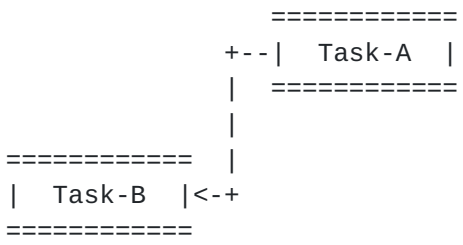

Figure 3: Start to finish relationship

RELTYPE=STARTTOSTART: The start of Task-A triggers the start of Task-B, that is Task-B can start anytime after Task-A starts.

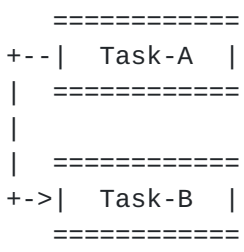

Figure 4: Start to start relationship

5. New Property Parameters

5.1. Split

Parameter name: SPLIT

Purpose: To indicate a series has been split.

Format Definition:

This parameter is defined by the following notation:

```
splitparam     = "SPLIT" "="
("YES"      ; The series is split
/ "NO"      ; The series is not split (default)
/ x-name        ; Experimental reference type
/ iana-token)   ; Other IANA registered type
```

Description: This parameter MAY be specified on the SRULE property to indicate that the series has been split with SPLIT=YES. Once split is is probably innapropriate to modify the series further.

5.2. Lookahead count

Parameter name: LOOKAHEAD-COUNT

Purpose: To specify the number of series instances that should be generated in advance.

Format Definition:

This parameter is defined by the following notation:

```
lookahead-countparam      = "LOOKAHEAD-COUNT" "=" 1*DIGIT
```

Description: This parameter MAY be specified on the SRULE property to indicate how many series instances should be generated in advance.

An implementation is free to apply its own limts but MUST NOT generate more than those defined by this parameter and/or the LOOKAHEAD-PERIOD parameter.

If both the LOOKAHEAD-PERIOD and LOOKAHEAD-COUNT arameters are supplied the result should be limited by both.

For example, if the LOOKAHEAD-PERIOD parameter would cause 8 instances to be generated but LOOKAHEAD-COUNT specifies 4 then only 4 instances will be generated.

5.3. Lookahead period

Parameter name: LOOKAHEAD-PERIOD

Purpose: To specify a maximum period for which series instances should be generated in advance.

Format Definition:

This parameter is defined by the following notation:

```
lookahead-periodparam      = "LOOKAHEAD-PERIOD" "="
                        DQUOTE dur-value DQUOTE
```

Description: This parameter MAY be specified on the SRULE property to indicate how far in advance series instances should be generated.

An implementation is free to apply its own limts but MUST NOT generate more than those defined by this parameter and/or the LOOKAHEAD-COUNT parameter.

If both the LOOKAHEAD-PERIOD and LOOKAHEAD-COUNT arameters are supplied the result should be limited by both.

For example, if the LOOKAHEAD-PERIOD parameter would cause 8 instances to be generated but LOOKAHEAD-COUNT specifies 4 then only 4 instances will be generated.

The value is a quoted duration.

6. New Properties

The SERIES-ID, LAST-SERIES-ID, SDATE and SXDATE properties are identical in form and in the parameters they take.

All must conform in form to the DTSTART property of the master component. Only the SDATE may specify a time which is not part of the calculated series.

The SRULE property value is identical in form to the RRULE property defined in [RFC5545]. The LOOKAHEAD-COUNT and LOOKAHEAD-PERIOD parameters indicate how many instances should be generated in advance.

6.1. Generating Series members

An agent, either the server or a client, will periodically extend the set of instances. The number of such generated instances is limited by:

Elements of the rule: The UNTIL or COUNT parts of the rule define when the series terminates. Thus a COUNT=100 specifies a maximum of 100 series members.

Lookahead count: This specifies how many series memerbs can exist from the current date/time. Thus a LOOKAHEAD-COUNT=4 means a maximum of 4 generated instances.

Lookahead period: This specifies how far into the future series members can be generated. Thus a LOOKAHEAD-PERIOD="PT2M" means a maximum period of 2 months.

System limits: This client or server SHOULD also apply limits to prevent a series from generating an overlarge set of members.

The starting point for the calculation is the DTSTART of the master component or the LAST-SERIES-ID if it exists in the master. In both cases the instance represented by that date is NOT generated as part of the intance set and the actual instance may have been excluded by an SXDATE property but the starting date is still valid.

The starting date/time property defines the first instance in the next batch of series members. Note that the starting property value MUST match the pattern of the series rule, if specified. For example, if the rule specifies every Wednesday the starting date MUST be a Wednesday.

The end date/time of the set will be provided by the UNTIL part of the rule, the LOOKAHEAD-PERIOD or by a system maxima.

A set of date/time values can be generated within those contraints. As each date/time value is generated it can be ignored if it is one of the SXDATE values.

Generation of values can terminate when the size of the result exceeds that given by the COUNT rule element, the LOOKAHEAD-COUNT value or any systm limit.

Any SDATE values that fall within the current range and are not in the set of SXDATE values can be added and the result truncated again to match the size limits.

Finally, any date/time values that have already been generated and are present as SERIES-ID values should be removed from the set. What remains is the new set of members to extend the current series.

The last of those values becomes the new value for the LAST-SERIES-ID property in the series master.

As noted above the "SXDATE" property can be used to exclude the value specified in the master. This leads to a complication as the master needs to be preserved as a container for the values which define the series. This is flagged by adding a DELETED-MASTER element to the SERIES-STATUS property.

6.2. Series UID

Property name: SERIES-UID

Purpose: This property defines the persistent, globally unique identifier for the full series.

Value Type: TEXT

Property Parameters: IANA and non-standard property parameters can be specified on this property.

Conformance: This property MUST be specified in any "VEVENT", "VTODO", and "VJOURNAL" calendar components acting as a series master or series instance.

Description: The SERIES-UID MUST be globally unique. This value SHOULD be generated by following the recommendations in section 5.3 of [RFC7986].

Format Definition:

This property is defined by the following notation:

```
seruid        = "SERIES-UID" seruidparam ":" text CRLF

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

Example:

The following is an example of this property:

```
SERIES-UID:123e4567-e89b-12d3-a456-426655440000
```

6.3. Series-exception-date

Property name: SXDATE

Purpose: This property defines the list of DATE-TIME exceptions for series of events, to-dos or journal entries.

Value Type: The default value type for this property is DATE-TIME. The value type can be set to DATE.

Property Parameters: IANA, non-standard, value data type, and time zone identifier property parameters can be specified on this property.

Conformance: This property can be specified in "VEVENT", "VTODO", and "VJOURNAL" calendar components acting as the series master.

Description: The exception dates, if specified, are used when computing the instances of the series. They specify date/time values which are to be removed from the set of possible series instances.

Format Definition:

This property is defined by the following notation:

```
  sxdate     = "SXDATE" sxdtparam ":" sxdtval *("," sxdtval) CRLF

  sxdtparam  = *(
              ;
              ; The following are OPTIONAL,
              ; but MUST NOT occur more than once.
              ;
              (";" "VALUE" "=" ("DATE-TIME" / "DATE")) /
              ;
              (";" tzidparam) /
              ;
              ; The following is OPTIONAL,
              ; and MAY occur more than once.
              ;
              (";" other-param)
              ;
              )

  sxdtval    = date-time / date
              ;Value MUST match value type
```

Example:

The following is an example of this property:

```
  SXDATE:19960402T010000Z,19960403T010000Z,19960404T010000Z
```

6.4. Series-date

Property name: SDATE

Purpose: This property defines the list of DATE-TIME values for series of events, to-dos or journal entries.

Value Type: The default value type for this property is DATE-TIME. The value type can be set to DATE.

Property Parameters: IANA, non-standard, value data type, and time zone identifier property parameters can be specified on this property.

Conformance: This property can be specified in "VEVENT", "VTODO", and "VJOURNAL" calendar components acting as the series master.

Description: This property can appear along with the "SRULE" property to define a extra series occurrences. When they both appear in a series master component, the instances are defined by the union of occurrences defined by both the "SDATE" and "SRULE".

Purpose:

This property is defined by the following notation:

```
  sdate      = "SDATE" sdtparam ":" sdtval *("," sdtval) CRLF

  sdtparam   = *(
              ;
              ; The following are OPTIONAL,
              ; but MUST NOT occur more than once.
              ;
              (";" "VALUE" "=" ("DATE-TIME" / "DATE" / "PERIOD")) /
              (";" tzidparam) /
              ;
              ; The following is OPTIONAL,
              ; and MAY occur more than once.
              ;
              (";" other-param)
              ;
              )

  sdtval     = date-time / date
               ;Value MUST match value type
```

Example:

The following are examples of this property:

```
  SDATE:19970714T123000Z
  SDATE;TZID=America/New_York:19970714T083000

  SDATE;VALUE=PERIOD:19960403T020000Z/19960403T040000Z,
   19960404T010000Z/PT3H

  SDATE;VALUE=DATE:19970101,19970120,19970217,19970421
   19970526,19970704,19970901,19971014,19971128,19971129,19971225
```

6.5. Series-id

Property name: SERIES-ID

Purpose: This property is used in conjunction with the "UID" and "SEQUENCE" properties to identify a specific instance of a "VEVENT", "VTODO", or "VJOURNAL" calendar component in a series. The property value is the original value of the "DTSTART" property of the series instance before any changes occur.

Value type: The default value type is DATE-TIME. The value type can be set to a DATE value type. This property MUST have the same value type as the "DTSTART" property contained within the series component. Furthermore, this property MUST be specified as a date with local time if and only if the "DTSTART" property contained within the series component is specified as a date with local time.

Property Parameters: IANA, non-standard, value data type and time zone identifier parameters can be specified on this property.

Conformance: This property can be specified zero or more times in any iCalendar component.

Description: The SERIES-ID is the originally calculated value of the DTSTART property based on the master identified by the RELATED-TO property with a RELTYPE=SERIES-MASTER parameter.

The full series of components can only be retrieved by searching for all components with a matching RELATED-TO property.

If the value of the "DTSTART" property is a DATE type value, then the value MUST be the calendar date for the series instance.

The DATE-TIME value is set to the time when the original series instance would occur; meaning that if the intent is to change a Friday meeting to Thursday, the DATE-TIME is still set to the original Friday meeting.

The "SERIES-ID" property is used in conjunction with the "UID" and "SEQUENCE" properties to identify a particular instance of an event, to-do, or journal in the series. For a given pair of "UID" and "SEQUENCE" property values, the "SERIES-ID" value for a series instance is fixed.

Format Definition:

This property is defined by the following notation:

```
  serid    = "SERIES-ID" sidparam ":" sidval CRLF

  sidparam   = *(
                   ;
                   ; The following are OPTIONAL,
                   ; but MUST NOT occur more than once.
                   ;
                   (";" "VALUE" "=" ("DATE-TIME" / "DATE")) /
                   (";" tzidparam) /
                   ;
                   ; The following is OPTIONAL,
                   ; and MAY occur more than once.
                   ;
                   (";" other-param)
                   ;
                   )

  sidval     = date-time / date
                   ;Value MUST match value type
```

Example:

The following are examples of this property:

```
  SERIES-ID;VALUE=DATE:19960401

  SERIES-ID;TZID=America/New_York:20170120T120000
```

6.6. Last series id

Property name: LAST-SERIES-ID

Purpose: To specify the last calculated instance of the series. When new instances are created they MUST have a SERIES-ID after the value of this property.

In all respects this property is identical to SERIES-ID and is in fact a copy of the SERIES-ID which would be present in the last created instance (assuming it is not suppressed by an SXDATE).

Value type: DATE or DATE_TIME (the default). This has the same requirements as SERIES-ID.

Property Parameters: IANA, non-standard, value data type and time zone identifier parameters can be specified on this property.

Conformance: This property MAY be specified in any iCalendar component.

Description: This property allows a client or server to delete the trailing instances in a series without having them recreated by some other agent.

Format Definition:

This property is defined by the following notation:

```
  last-series-i   = "LAST-SERIES-ID" lastseriesidparam  /
                    (
                      ";" "VALUE" "=" "TEXT"
                      ":" text
                    )
                    (
                      ";" "VALUE" "=" "REFERENCE"
                      ":" text
                    )
                    (
                      ";" "VALUE" "=" "URI"
                      ":" uri
                    )
                    CRLF


  lastseriesidparam = *(

                   ; the following is MANDATORY
                   ; and MAY occur more than once

                   (";" relparam) /

                   ; the following are MANDATORY
                   ; but MUST NOT occur more than once

                   (";" fmttypeparam) /
                   (";" labelparam) /
                   ; labelparam is defined in ...

                   ; the following is OPTIONAL
                   ; and MAY occur more than once

                   (";" xparam)

                   )
```

Example:

The following is an example of this property. It points to a server acting as the source for the calendar object.

```
LINK;REL=SOURCE;LABEL=The Egg:http://example.com/events
```

6.7. Series Rule

Property name: SRULE

Purpose: This property defines a rule or repeating pattern for a series of events, to-dos or journal entries.

Value Type: RECUR

Property Parameters: IANA, non-standard, look-ahead count or date property parameters can be specified on this property.

Conformance: This property can be specified in any "VEVENT", "VTODO", and "VJOURNAL" calendar component.

[RFC5545] states that the RRULE component SHOULD only appear once. This is unduly limiting for those applications that require more than one rule to specify the series. In fact, some complex rules are better expressed as multiple rules. Recipients of iCalendar data containing SRULE MUST support the occurrence of multiple SRULEs.

Clients and servers MAY choose to reduce the number of SRULE properties to 1. In reality many - if not most - clients are unable to handle the full set of SRULE or RRULE features.

Description: The series rule, if specified, is used in computing the instances to be generated for the series. These are generated by considering the master "DTSTART" property along with the "SRULE", "SDATE", and "SXDATE" properties contained within the series master. The "DTSTART" property defines the first instance in the recurrence set which is represented by that master event.

Unlike the RRULE the "DTSTART" property MUST be synchronized with the series rule, if specified. For example, if the DTSTART specifies a date on Wednesday but the SRULE specifies every Tuesday then a server or client MUST reject the component.

The final series is represented by gathering all of the start DATE-TIME values generated by any of the specified "SRULE" and "SDATE" properties, and then excluding any start DATE-TIME values

specified by "SXDATE" properties. This implies that start DATE-TIME values specified by "SXDATE" properties take precedence over those specified by inclusion properties (i.e., "SDATE" and "SRULE"). Where duplicate instances are generated by the "SRULE" and "SDATE" properties, only one instance is considered. Duplicate instances are ignored.

The "DTSTART" property specified within the master iCalendar object defines the first instance of the recurrence. In most cases, a "DTSTART" property of DATE-TIME value type used with a series rule, should be specified as a date with local time and time zone reference to make sure all the recurrence instances start at the same local time regardless of time zone changes.

If the duration of the series component is specified with the "DTEND" or "DUE" property, then the same exact duration will apply to all the members of the generated series. Else, if the duration of the series master component is specified with the "DURATION" property, then the same nominal duration will apply to all the members of the generated series and the exact duration of each instance will depend on its specific start time. For example, series instances of a nominal duration of one day will have an exact duration of more or less than 24 hours on a day where a time zone shift occurs. The duration of a specific instance may be modified in an exception component or simply by using an "SDATE" property of PERIOD value type.

Format Definition:

This property is defined by the following notation:

```
  srule      = "SRULE" srulparam ":" recur CRLF

  sruleparam = *(
                ; the following are OPTIONAL
                ; but MUST NOT occur more than once

                (";" lookahead-countparam) /
                (";" lookahead-periodparam) /

                ; the following is OPTIONAL
                ; and MAY occur more than once

                (";" xparam)

                )
```

Examples: Refer to [RFC5545] for example of the use of RRULE which has identical format.

6.8. SITEM-STATUS Property

Property name: SITEM-STATUS

Purpose: This property is used to define the status of a series item.

Conformance: This property MAY be specified in any iCalendar series item.

Description: An item may be generated to provide an indication to a user or client of when an event may occur - even if that event is outside of the generated period.

When this property is absent the series item is treated as a concrete item which may be updated subject to the constraints of the server/storage for the item.

Format Definition:

This property is defined by the following notation:

```
 sitem-status    = "SITEM-STATUS" sisparam ":" SISvalue CRLF

 sisparam   = *(";" other-param)

 sisvalue = "DUMMY"  ; This item is generated as an indication of where
                ; an item might appear when it is generated
              / iana-token
              / x-name
```

Example:

The following is an example of this property.

```
    SITEM-STATUS:DUMMY
```

7. Redefined RELATED-TO Property

7.1. RELATED-TO

Property name: RELATED-TO

Purpose: This property is used to represent a relationship or reference between one calendar component and others. The

definition here extends the definition in Section 3.8.4.5. of [RFC5545] by including a section on RELTYPE=SERIES-MASTER.

Value type: URI, UID or TEXT

Property Parameters: Relationship type, IANA and non-standard property parameters can be specified on this property.

Conformance: This property MAY be specified in any iCalendar component.

Description: By default or when VALUE=UID is specified, the property value consists of the persistent, globally unique identifier of another calendar component. This value would be represented in a calendar component by the "UID" property.

By default, the property value points to another calendar component that has a PARENT relationship to the referencing object. The "RELTYPE" property parameter is used to either explicitly state the default PARENT relationship type to the referenced calendar component or to override the default PARENT relationship type and specify either a CHILD or SIBLING relationship or a temporal relationship.

The PARENT relationship indicates that the calendar component is a subordinate of the referenced calendar component. The CHILD relationship indicates that the calendar component is a superior of the referenced calendar component. The SIBLING relationship indicates that the calendar component is a peer of the referenced calendar component.

The FINISHTOSTART, FINISHTOFINISH, STARTTOFINISH or STARTTOSTART relationships define temporal relationships as specified in the reltype parameter definition.

The SERIES-MASTER relationship when included in a series instance refers to the master of that series. When included in a series master it refers to a previous master in a chain of spilt series.

Changes to a calendar component referenced by this property can have an implicit impact on the related calendar component. For example, if a group event changes its start or end date or time, then the related, dependent events will need to have their start and end dates changed in a corresponding way. Similarly, if a PARENT calendar component is cancelled or deleted, then there is an implied impact to the related CHILD calendar components. This property is intended only to provide information on the relationship of calendar components. It is up to the target

calendar system to maintain any property implications of this relationship.

Format Definition:

This property is defined by the following notation:

```
related    = "RELATED-TO" relparam ( ":" text ) /
(
";" "VALUE" "=" "UID"
":" uid
)
(
";" "VALUE" "=" "URI"
":" uri
)
CRLF

relparam   = *(
;
; The following are OPTIONAL,
; but MUST NOT occur more than once.
;
(";" reltypeparam) /
(";" gapparam) /
;
; The following is OPTIONAL,
; and MAY occur more than once.
;
(";" other-param)
;
)
```

Example:

The following are examples of this property.

```
RELATED-TO;RELTYPE=SERIES-MASTER:19960401-080045-4000F192713
```

8. Backwards compatibility

Any clients following the approach specified in [RFC5545] are expected to ignore any properties, parameters or components they don't recognize.

For such clients the series appears to be an unconnected set of components. They all have their own unique UIDS. If the client

updates an instance this should be identical in effect to an update carried out by a client aware of the new properties.

Updates MUST preserve the SERIES-ID, LAST-SERIES-ID, SRULE, SDATE and SXDATE properties. A client which does not do so is in violation of [RFC5545].

There are two possible problem areas: first we must prevent series unaware clients from updating the masters and secondly we must prevent attempts to update dummy instances.

Both case are handled by the definition of a new header field : "ICAL-SERIES". The presence of this header field in requests from a client indicates that it is aware of this specification. In the absence of the header servers MUST NOT send master template items.

In the case of an absent header servers MUST refuse to accept updates to dummy items - e.g. with an HTTP forbidden status in CalDAV.

9. CalDAV extensions

This specification may extend Caldav by adding reports to return all members of a series given the series master UID. This could be handled by the current query mechanism but it is likely to be sufficiently frequently used that a special query is appropriate.

It is also likely we will want a CalDAV operation to split a series and generate the additional members of the series as a single atomic operation.

10. Security Considerations

Clients and servers should take care to limit the number of generated instances to a reasonable value. This can be a relatively small value.

11. IANA Considerations

11.1. iCalendar Property Registrations

The following iCalendar property names have been added to the iCalendar Properties Registry defined in Section 8.3.2 of [RFC5545]

Property	Status	Reference
LAST-SERIES-ID	Current	Section 6.6
SERIES-ID	Current	Section 6.5
SERIES-UID	Current	Section 6.2
SDATE	Current	Section 6.4
SRULE	Current	Section 6.7
SXDATE	Current	Section 6.3

11.2. iCalendar Property Parameter Registrations

The following iCalendar property parameter names have been added to the iCalendar Parameters Registry defined in Section 8.3.3 of [RFC5545]

Parameter	Status	Reference
LOOKAHEAD-COUNT	Current	Section 5.2
LOOKAHEAD-PERIOD	Current	Section 5.3
SPLIT	Current	Section 5.1

11.3. iCalendar RELTYPE Value Registrations

The following iCalendar "RELTYPE" values have been added to the iCalendar Relationship Types Registry defined in Section 8.3.8 of [RFC5545]

Relationship Type	Status	Reference
SERIES-ID	Current	Section 4

12. Acknowledgements

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

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

13. Normative References

[I-D.daboo-caldav-attachments]
Daboo, C. and A. Quillaud, "CalDAV Managed Attachments", draft-daboo-caldav-attachments-03 (work in progress), February 2014.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/info/rfc3986>.

[RFC5545] Desruisseaux, B., Ed., "Internet Calendaring and Scheduling Core Object Specification (iCalendar)", RFC 5545, DOI 10.17487/RFC5545, September 2009, <https://www.rfc-editor.org/info/rfc5545>.

[RFC5988] Nottingham, M., "Web Linking", RFC 5988, DOI 10.17487/RFC5988, October 2010, <https://www.rfc-editor.org/info/rfc5988>.

[RFC7986] Daboo, C., "New Properties for iCalendar", RFC 7986, DOI 10.17487/RFC7986, October 2016, <https://www.rfc-editor.org/info/rfc7986>.

[W3C.REC-xml-20060816]
Bray, T., Paoli, J., Sperberg-McQueen, M., Maler, E., and F. Yergeau, "Extensible Markup Language (XML) 1.0 (Fourth Edition)", World Wide Web Consortium Recommendation REC-xml-20060816, August 2006, <http://www.w3.org/TR/2006/REC-xml-20060816>.

[W3C.WD-xptr-xpointer-20021219]
DeRose, S., Daniel, R., and E. Maler, "XPointer xpointer() Scheme", World Wide Web Consortium WD WD-xptr-xpointer-20021219, December 2002, <http://www.w3.org/TR/2002/WD-xptr-xpointer-20021219>.

Appendix A. Points for discussion

Splitting and linking: The spec currently only allows for backward linking to previous masters. There is a parameter added to the rule SPLIT=YES to indicate that the series was split

It makes sense to have a forward link to the new(er) series. However, a client/server may not know what the UID is until after data is stored. The new chain can be determined via a query so perhaps we can leave it up to the protocols to figure out that mechanism.

CalDAV queries: if there were a better more generalised query language such an extensions might be unnecessary. Should we define a query language specifically for calendaring?

Appendix B. Change log

2018-01-01 MD Better descriptions - LAST-SESSION-ID.

2017-09-30 MD Minor updates: better backward compatibility.

2017-02-12 MD Initial version

Author's Address

Michael Douglass
Spherical Cow Group
226 3rd Street
Troy, NY 12180
USA

Email: mdouglass@sphericalcowgroup.com
URI: http://sphericalcowgroup.com