Workgroup: Network Working Group
Internet-Draft: draft-apthorp-ical-tasks-02
Updates: 5545 (if approved)
Published: 9 May 2021
Intended Status: Standards Track
Expires: 10 November 2021
Authors: A. Apthorp   C. Daboo
DHL Express   Apple Inc.
M. Douglass
Bedework Commercial Services

Task Extensions to iCalendar

Abstract

This document defines extensions to the Internet Calendaring and Scheduling Core Object Specification (iCalendar) (RFC5545) to provide improved status tracking, scheduling and specification of tasks.

It also defines how Calendaring Extensions to WebDAV (CalDAV) (RFC 4791) servers can be extended to support certain automated task management behaviours.

Status of This Memo

Copyright Notice

Table of Contents

1. Acknowledgements

The authors 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:

John Chaffee, Marten Gajda, Ken Murchison

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

2. Introduction

This document specifies extensions to the existing Internet Calendaring and Scheduling Core Object Specification (iCalendar) [RFC5545], and associated protocols, in order to enhance the structured communication and execution of tasks. The enhancements allow for the communication, time planning and scheduling of tasks by and between automated systems (e.g. in smart power grids, business process management systems) as well as for human centered tasks.

A "task" is a representation of an item of work assigned to an individual or organization. In the iCalendar Object Model [RFC5545] the representation of tasks is by "VTODO" calendar components. Tasks can be identified in a number of situations, either informally as ad-hoc tasks in personal "to-do" lists or more formally in:

*Business processes - ranging from repetitive workflows to adaptive cases and trouble ticketing

*Project Management - whether for large scale construction projects or collaborative software development

The extensions specified here are defined in the context of an overall architecture for task calendaring and scheduling.

2.1. Terms and Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Terms defined in this specification include:

Assignee A calendar user assigned to perform a given task. An assignee is equivalent to an attendee of an event.

Calendar User (CU) A person or software system that accesses or modifies calendar information.

Calendar User Agent (CUA) This may be

1. Software with which the calendar user communicates with a calendar service or local calendar store to access calendar information.

2. Software that gathers calendar data on the Calendar User's behalf.

Candidate A calendar user who might be able to perform a given task, prior to actually being assigned the task, e.g., a dispatcher has a list of taxi drivers (candidates) from which one will be selected to pick-up a passenger.

Organizer A calendar user who creates a calendar item, requests free/busy information, or published free/busy information. It is an Organizer who invites Attendees [RFC5545].

Observer A calendar user interested in a calendar component, e.g., a manager may have interest in all tasks that have not been completed.

Resource A resource in the scheduling context is any shared entity that can be scheduled by a calendar user, but does not control its own attendance status. Resources can be of "Location", "Equipment", or "Role" type.

Task A representation of an item of work that can be assigned to one or more task actor assignees. In [RFC5545], these are "VTODO" calendar components, which are groupings of component properties and possibly "VALARM" calendar components that represent an action-item or assignment.

3. Task Architecture

A reference architecture for task calendaring and scheduling is defined in order to identify the key logical elements involved in task management and the interfaces between them to enable interoperability. The logical elements identified here establish an appropriate separation of concerns and clarify the responsibilities of different elements. However, the architecture does not prescribe a binding or packaging of elements, i.e., software systems may be developed where some elements are tightly bound and the interfaces between bound elements are not exposed. The task architecture is also described in [TARCH].

```
  Task        +-------+
  Trigger             |
+---------------------V---------------------+    +-----------+
|           Task Generating System          |    |           |
|        +-------------------------+        |    |           |
|        |            O            |        |    |           |
|        |           /|\           |        |    |           |
|        |           / \           |        |    |           |
|        |      Task Organizer     |        <---->           |
|        +-^--------^--------------+        |    |           |
|          |        |                       |    |           |
| +--------V-+ +----V-----+    +----------+ |    |           |
| |   Task   | | Process  |    |   Task   | |    |           |
| |Assignment| |  Logic   <---->  Domain  | |    |           |
| |  Rules   | |          |    |   Data   | |    |           |
| +----------+ +----------+    +----------+ |    |           |
|                                           |    |           |
+------^----------+-----^-------------------+    |           |
       |          |     |                        |           |
  Availability  Task   Task                      |           |
       |          |   Status                     |           |
       |          |     |                        |           |
+------v----------v-----+-------------------+    |           |
|      Calendar and Scheduling System       |    | Directory |
| +---------+  +---------+                  |    |  Service  |
| |         |  |  Task   |                  <---->           |
| |Schedule |  |  Lists  |                  |    |           |
| |         |  |         |                  |    |           |
| +---------+  +---------+      Server      |    |           |
+-------------------------------------------+    |           |
|                               Client      |    |           |
| +----------------------+    +-----------+ |    |           |
| |       Calendar       |    |   Task    | |    |           |
| |      User Agent      +----> Specific  | <---->           |
| |                      |    |Application| |    |           |
| +----------------------+    +-----------+ |    |           |
|                                           |    |           |
+-----^---------^--------+---------+--------+    |           |
      |         |        |         |             |           |
+-----V---------V--------V---------V--------+    |           |
|                Task Actors                |    |           |
|     O         O        O         O        |    |           |
|    /|\       /|\      /|\       /|\       +---->           |
|    / \       / \      / \       / \       |    |           |
|          Candidate(s)        Observer(s)  |    |           |
| Assignee(s)       Resource(s)             |    |           |
+-------------------------------------------+    +-----------+
```

4. Task Architecture Elements

The following logical elements form the task architecture that this specification is based on:

Task Actors Various calendar users that may be involved in the monitoring or performing of a task. The set of actors includes: Organizers, Observers, Resources, Assignees, and Candidates.

Task Organizer The Organizer of a task.

Task Domain Data This is any domain specific data that may be acted on or provides context to it in performing a task.

Task Specific Application A task specific application renders the data concerning the task (including task domain data) for presentation and manipulation by a task actor.

Process Logic Determines under what conditions a task (or tasks) is generated and the actions to take on completion, or some other status event occurring (or not) on the task.

Task Trigger This is some event that gives rise to the generation of a task according to Process Logic. Task triggers can come from many different sources including, for example; a task being requested through the calendaring system, a status change in the progression of a business process being managed by a business process management or ERP system.

Task Assignment Rules Govern how actors are assigned to a task. A range of different assignment patterns [WfRP] may be considered, including the two general cases:

1. Delegation to a named actor or group of actors

2. Advertising to a pool of actors for self-selection

In either case the assignment may be made based on a variety of criteria including, name, availability, skills, capacity, etc.

Task Generating System A system that creates and assigns tasks in response to some initiating event (task trigger). Task creation is according to Process Logic with task assignment determined by Task Assignment Rules. This system also tracks the status of tasks and will initiate further actions based upon the status. A task generating system can take many forms, for example; Business Process Management System, Project Management System, Bug Tracking System, Building Control System. A Task Generating System may also be a human. In iCalendar terms the Task Generating System is the organizer.

Human Task Generation
Task creation, assignment and tracking coordinated by a human organizer is a special case of a task generating system. In this case Task Assignment Rules and Process Logic may be either explicit or tacit.

Directory Service A software system that stores and provides access to information providing details of task actors that may participate or be interested in a task.

Calendar and Scheduling System A software system that stores, publishes and synchronizes calendar data such as events, tasks and journal entries for actors. In the context of tasks this includes schedules (i.e. allocated time and availability to perform tasks) and task lists. A calendar and scheduling system typically consists of server and client software components.

It is not within the scope of this document to specify how Process Logic or Task Assignment Rules are codified. Such logic and rules may be codified in a variety of ways, including traditional programming languages (e.g. C++, Java) or process modelling languages (e.g. BPMN [BPMN]).

5. Architecture Foundations

The key standards that enable interoperability between the logical elements of the architecture are the Internet Calendaring and Scheduling Core Object Specification (iCalendar) [RFC5545] and associated protocols. Task and task status are represented by the iCalendar "VTODO" component. Protocols include, in particular, the iCalendar Transport-Independent Interoperability Protocol (iTIP) [RFC5546] for task assignment and scheduling, and Calendaring Extensions to WebDAV (CalDAV) [RFC4791] for client server communication.

Additionally, this specification uses definitions from Support for iCalendar Relationships [I-D.ietf-calext-ical-relations]. The LINK, REFID, RELATED-TO and CONCEPT properties enable context and a rich set of relationships between tasks and other iCalendar components to be specified.

6. Task Extensions

In order to support the task architecture described in Section 3, this document defines a number of extensions to the current iCalendar standards in the areas of:

Task Specification improved ability to specify domain specific tasks

Task Deadlines, Milestones and Time Planning clarification of deadlines and extension for task duration to support task time planning

Task Scheduling and Assignment ensure support for common pattens of scheduling and assigning tasks

Task Status Tracking improved granularity in status tracking information and alerting task actors to pending or actual task status changes

These extensions are supported mainly by additions to the properties and parameters used within the "VTODO" component.

7. Task Specification

The specification of tasks must be semantically explicit in order for them to be managed within the context of a business process or project, and be understood by both humans and IT systems. The current VTODO component only provides for simple ad-hoc tasks or 'to do' lists, and is therefore extended by this specification as follows:

Task type explicitly what type of task is to be performed is identified.

Task context and relationships how a specific task relates to other tasks and other objects that need to be understood for the effective execution of a task.

Task specific data the form and content of domain data provided as input to a task and/or that may be output from a task.

Organizer and attendee recognizes that a task organizer or attendee can be an automated system.

7.1. CONCEPT for task type identification

The CONCEPT property is used to identify the type of task, for example;

```
CONCEPT:http://example.com/task/delivery
```

7.2. Task Context and Relationships

The LINK property specifies a link to external information, which may be context to the task. For example:

```
LINK;REL=SOURCE:http://example.com/package/1234567890

LINK;REL=describedby:mid:752142.1414823874.307E5@mx123.example.com
```

The external information may be data to be manipulated in performing the task. See section 3.1.3 Task Domain Data Handling.

REFID is used to identify a key allowing the association of tasks that are related to the same object and retrieval of a task based on this key. This may be, for example, to identify the tasks associated with a given project without having to communicate the task structure of the project, or all tasks associated to a specific package.

```
REFID:Manhattan

REFID:1234567890
```

Extensions [Doug114] to the RELATED-TO property allow temporal relationships between tasks as found in project management to be specified as well as parent / child relationships and dependencies (DEPENDS-ON). Tasks (VTODOs) may also be related to other calendar components; for example to a VEVENT to block time to perform a task.

7.3. Task Domain Data Handling

Provide support for task specific input and output data (including updates) beyond the standard iCalendar properties. It is envisaged that standard calendar user agents will be able to launch task specific applications by passing task specific data.

The LINK property can be used to 'attach' the domain specific data to the task. For example, it might be a URI pointing to a web page where the status of the task can be directly manipulated.

```
LINK;REL="vacation-system";VALUE=URI:http://example.com/
vacation-approval?id=1234
```

Or it might be used for attachments specific to the task, for example an electronic copy of a signature taken to confirm delivery of a package.

```
LINK;REL="electronic-signature";VALUE=URI:http://example.com/
delivery/sig1234.jpg
```

8. Task Deadlines, Milestones and Time Planning

Deadlines for starting and finishing a task are defined by the DTSTART, DUE and DURATION properties. DTSTART represents the earliest start time for beginning work on a task. DUE, or DTSTART + DURATION represent the latest finish time for a task. Thus these properties define a "window" within which a task has to be performed. However, there is currently no way to indicate how long the task is expected to take. This document defines a new property, ESTIMATED-DURATION, to allow the estimated time that a task should take to be specified separately from the deadlines for starting and finishing a task. This supports time planning by enabling calendar user agents to display when tasks should occur and therefore allow calendar users to visualize when tasks should be performed and allocate time to them.

A task that has intermediary deadlines (i.e., milestones) SHOULD be expressed by child VTODO components (i.e., sub-tasks associated with each of the milestones) in conjunction with the RELATED-TO property to relate the parent and child tasks.

9. Task Scheduling and Assignment

This specification supports the two distinct models of assigning actors to tasks, i.e., 1) strictly one assignee per task or 2) task assignment to multiple assignees. In this regard one or many ATTENDEES may be specified against a task depending upon the model applied by the task organizer.

In addition a number of different patterns of resource or assignee identification are anticipated. The specific Task Assignment Rules are the responsibility of the Task Organizer.

Communication of task assignment or delegation to one or more actors who are allocated to a task by the organizer is directly supported by iTIP, i.e., all included ATTENDEES in an iTIP REQUEST are expected to perform the task.

The offering or advertising of a task to one or more (potential) actors where only one or a subset of the candidates may accept the task will be addressed by a new VPOLL mode (See Appendix B) [VPOLL].

10. Status Reporting

10.1. Improved granularity in status reporting information

This document defines new status parameters that can be applied to the VTODO status (STATUS) property, as well as the participant status (PARTSTAT) parameter. These new parameters provide additional information on why (REASON) and when (MODIFIED) a status has changed. In addition to these parameters new status values are specified to provide for task suspension, failure and preparation.

10.2. Relating comments to status

The GROUP parameter is used with the STATUS or ATTENDEE properties to relate an associated COMMENT property. The COMMENT property can then be used to include additional human readable information about why the associated STATUS or ATTENDEE property changed.

```
STATUS;REASON="http://example.com/reason/delivery-failed";
 SUBSTATE=ERROR;MODIFIED=20130212T120000Z;GROUP=G1:FAILED
COMMENT;MODIFIED=20130226T110451Z;GROUP=G1:Breakdown
ATTENDEE;PARTSTAT=FAILED;MODIFIED=20130226T1104510Z;GROUP=G2:
REASON="http://example.com/reason/van-break-down":
 mailto:xxx@example.com
COMMENT;MODIFIED=20130226T110451Z;GROUP=G2:Puncture
```

10.3. Comments associated to reasons and status changes

Reasons may be associated directly with a comment, allowing for multiple reasons associated with a status to each have a comment associated with them [EDISTS].

```
CONCEPT:http://example.com/task/delivery
STATUS;SUBSTATE=ERROR;MODIFIED=20130212T120000Z;GROUP=G1:FAILED
COMMENT;MODIFIED=20130226T110451Z;GROUP=G1:Out of time
COMMENT;REASON="http://example.com/reason/traffic";
 MODIFIED=20130226T110451Z;GROUP=G1:Traffic Accident on E44
COMMENT;REASON="http://example.com/reason/closed";
 MODIFIED=20130226T110451Z;GROUP=G1:Arrived after office hours
```

10.4. Task Alerts and Notifications

Different needs to alert or notify task actors of pending or actual task status changes are recognized:

Alarms Alarms (VLARM components) operate in the calendar user agent space to notify the task actor of a pending task state for a task they are assigned to or are interested in. Note: there is no constraint in the current standards on the propagation of alarms specified on calendar objects by organizers to individual attendees.

Escalations An escalation or notification to the ATTENDEE, ORGANIZER, or other task actor may be required if a deadline associated with a task is exceeded or for some other reason. Process Logic identifying when and who to propagate escalations to is the responsibility of the Task Generating System, e.g., a BPMS.

Notifications Task actors (observers) not directly involved in performing a task but with a known interest in a given task's status can be identified by the ASSOCIATE property [Doug214] against certain components e.g. ALARM, to identify which task events the stakeholder/party is interested in. Notifications on shared calendars will allow task actors to register an interest in changes to tasks within a calendar (see Appendix A).

10.5. Automated Status Changes

A new property, TASK-MODE, is introduced to instruct servers to apply automated operations for changing the status of a task.

11. New Property Parameters

11.1. Group

Parameter name GROUP

Purpose This parameter allows the association of different (usually) multiply occurring properties.

Format Definition This parameter is defined by the following notation:

```
groupparam       = "GROUP" "=" text
                   *("," text)
```

Description The value of this parameter is free-form text that creates an identifier for associated properties. All properties that use the same GROUP value are associated through that value. For example, multiple comments and an attendee may be associated with a status value.

Example The following is an example of this property.

```
GROUP=G1
```

11.2. Reason

Parameter name REASON

Purpose To indicate the reason for a change in status of a task or attendee participation status.

Format Definition This parameter is defined by the following notation:

```
reasonparam      = "REASON" "=" DQUOTE uri DQUOTE
                   *("," DQUOTE uri DQUOTE)
```

Description This property parameter allows the change in status of a task or participant status to be qualified by the reason for the change with a codified reason. Typically reasons are defined within the context of the task type and therefore SHOULD include the name-space of the authority defining the task. Common reason codes are IANA registered and do not have a name-space prefix.

Example

```
STATUS;REASON="http://example.com/reason/delivered-on-time";
MODIFIED=20130212T120000Z;GROUP=G1:COMPLETED
ATTENDEE;REASON="x-example-reason:out-of-office";
 PARTSTAT=DECLINED;MODIFIED=20130212T120000Z;
 GROUP=123:mailto:cyrus@example.com
```

11.3. Modified

Parameter name MODIFIED

Purpose To specify the time and date of when the status of a task or attendee participant status changed.

Format Definition This parameter is defined by the following notation:

```
modifiedparam    = "MODIFIED" "=" date-time
```

Description The modified parameter allows the specification of the date time of when a status (STATUS) or participant status (PARTSTAT) changed. It MUST be specified in the UTC time format. The value of MODIFIED SHOULD be set at the time when the associated status (either STATUS or PARTSTAT)is changed.

Therefore either a client or server may set the value of MODIFIED depending on which is updating the value of STATUS or PARTSTAT. For backwards compatibility if the server detects that MODIFIED should have changed but wasn't (for example the client doesn't support MODIFIED) then the server MAY set MODIFIED retrospectively.

Example

```
STATUS;REASON=""http://example.com/reason/delivered-on-time";
 MODIFIED=20130212T120000Z;GROUP=G1:COMPLETED
```

11.4. Sub-State

Parameter name SUBSTATE

Purpose To provide additional granularity of task status for e.g. IN-PROCESS.

Format Definition This parameter is defined by the following notation:

```
substateparam    = "SUBSTATE" "="
                    ( "OK"        ; everything is fine(the default)
                    / "ERROR"     ; something is wrong (the REASON
                    ; code explains why)
                    / "SUSPENDED" ; waiting on some other task to
                    ; complete or availability of a
                    ; resource (REASON code explains
                    ; why)
                    / x-name      ; Experimental type
                    / iana-token) ; Other IANA-registered type
```

Description The sub-state parameter allows additional qualification and granularity of states to be recorded, in particular for the IN-PROCESS state. It allows individual sub-states to be recorded without the need to define and publish a sub-task associated with a parent task purely to track that a particular state has been reached. This property also allows parallel states to be expressed e.g. that a task has been suspended at whatever state it has reached.

Example STATUS;REASON="http://example.com/reason/no-one-home";

```
 SUBSTATE=ERROR:FAILED
STATUS;REASON="http://example.com/reason/paint-drying";
 SUBSTATE=SUSPENDED:IN-PROCESS
```

12. New Parameter Values

12.1. Redefined VTODO Participant Status

Participant status parameter type values are defined in [RFC5545]. This specification redefines that type to include the new value FAILED for VTODO iCalendar components.

Format Definition This property parameter is extended by the following notation:

```
partstat-todo    /= *("FAILED")  ; To-do cannot be completed
```

Example

```
ATTENDEE;REASON="http://example.com/reason/not-enough-time";
 PARTSTAT=FAILED:mailto:jsmith@example.com
```

13. New Properties

13.1. Estimated Duration

Property Name ESTIMATED-DURATION

Purpose This property specifies the estimated positive duration of time the corresponding task will take to complete.

Value Type DURATION

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

Conformance This property can be specified in "VTODO" calendar components.

Format Definition This property is defined by the following notation:

```
est-duration  = "ESTIMATED-DURATION" durparam ":" dur-value CRLF
                ;consisting of a positive duration of time.

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

Description

In a "VTODO" calendar component the property MAY be used to specify the estimated duration for the to-do, with or without an explicit time window in which the event should be started and completed. When present, DTSTART and DUE/DURATION represent the window in which the task can be performed. ESTIMATED-DURATION SHOULD be passed from ORGANIZER to ATTENDEE in iTIP [RFC5546] messages.

Example The following is an example of this property that specifies an interval of time of exactly one hour:

```
ESTIMATED-DURATION:PT1H
```

13.2. Task Mode

Property Name TASK-MODE

Purpose This property specifies automatic operations that servers apply to tasks based on changes in attendee status (PARTSTAT).

Value Type TEXT

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

Conformance This property can be specified zero or more times in a "VTODO" calendar component.

Format Definition This property is defined by the following notation:

```
task-mode   = "TASK-MODE taskmodeparam ":" taskvalue
              *("," taskvalue) CRLF

taskvalue   = "AUTOMATIC-COMPLETION" ; set STATUS completed
                ;if all attendees have completed
                / "AUTOMATIC-FAILURE"
                / "SERVER"
                / "CLIENT"
                / iana-token
                / x-name

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

Description In a "VTODO" calendar component this property MAY be used to indicate to servers how they can automatically change the state of the task based on iTIP replies from Attendees. For example, the server can automatically set the overall task status (STATUS) to COMPLETED when every attendee has marked their own status (PARTSTAT) as COMPLETED, or the server could mark the task as FAILED if its DUE date passes without it being completed. TASK-MODE processing is performed on the organizer's copy of the task.

The property value is a list of one or more IANA registered tokens that defines modes to be used for the task. This specification defines three modes which are described in the following sub-sections.

Examples

```
        TASK-MODE:AUTOMATIC-COMPLETION,AUTOMATIC-FAILURE
TASK-MODE:SERVER
TASK-MODE:AUTOMATIC-FAILURE
```

AUTOMATIC-COMPLETION Task Mode The task mode value "AUTOMATIC-COMPLETION" indicates to the server that it can change the "VTODO" component's STATUS property value to "COMPLETED" as soon as all ATTENDEEs in the task have replied with a "PARTSTAT" parameter set to "COMPLETED".

AUTOMATIC-FAILURE Task Mode The task mode value "AUTOMATIC-FAILURE" indicates to the server that it SHOULD change the "VTODO" component's STATUS property value to "FAILED" if either:

*the PARTSTAT of one ATTENDEE is set to FAILED; or

*the current time is past the effective due date of the component and the task has not yet been completed.

Note: The effective due date is either the "DUE" property value or the combination of the "DTSTART" and "DURATION" property values.

CLIENT Task Mode The task mode value "CLIENT" is an instruction to the server to honour the status set by the client.

SERVER Task Mode The task mode value "SERVER" indicates to the server that it can change the "VTODO" component's STATUS property value to an appropriate value, based on implementation defined "business rules", as ATTENDEE responses are processed or as deadlines related to the task pass.
The server can add this property to a "VTODO" component to indicate to the client that it will be managing the status.

14. Property Extensions and Clarifications

14.1. The ATTENDEE property

The Attendee property is defined in [RFC5545]. This specification extends that property to include new parameters to indicate the reason for a participant status change (See Appendix A) and sub-states.

Format Definition This property is defined by the following notation:

```
attendee   = "ATTENDEE" attparam ":" cal-address CRLF

attparam   /= *(
                ;
                ; The following are OPTIONAL,
                ; but MUST NOT occur more than once.
                ;
                (";" reasonparam)
                (";" modifiedparam)
                (";" substateparam)
                )
```

Example: The following are examples of this property's use for tasks:

```
ATTENDEE;PARTSTAT=DECLINED;MODIFIED=20130212T120000Z;GROUP=G1;
 REASON="http://example.com/reason/too-busy":mailto:xxx@example.com

ATTENDEE;PARTSTAT=IN-PROCESS;MODIFIED=20130212T120000Z;
 SUBSTATE=X-EXAMPLE-STEP-1:mailto:xxx@example.com
```

14.2. Redefined COMMENT Property Parameter List

The Comment property is defined in [RFC5545].

Format Definition The "COMMENT" property parameter list is augmented as follows:

```
commparam    /= *(
                ; The following are OPTIONAL,
                ; but MUST NOT occur more than once.
                (";" reasonparam) /
                (";" modifiedparam)
                )
```

14.3. Redefined STATUS Property

The Status property is defined in [RFC5545]. This specification extends that property to include new parameters to indicate the reason for a status change as well as new values associated with VTODO iCalendar components (See Appendix A for examples of the task state lifecycle).

Format Definition The "STATUS" property parameter list is augmented as follows:

```
statparam      /= *(
                    ; The following are OPTIONAL,
                    ; but MUST NOT occur more than once.
                    ;
                    (";" reasonparam)
                    (";" modifiedparam)
                    (";" substateparam) /
                    )

statvalue-todo = / "PENDING"    ;Indicates a to-do has been
                ;created and accepted, but has not
                ;yet started.
                / "FAILED"       ;Indicates to-do has failed.
                ;Extended status values for
                ;"VTODO".
```

Description:

PENDING - A task has been created but has not yet started and is ready to start subject to other dependencies (e.g. preceding task or DTSTART). This is the default state.

FAILED - task has failed and may need some follow-up from the organizer to re-schedule or cancel

Example: The following is an example of this property for a "VTODO" calendar component:

```
STATUS;REASON="http://example.com/reason/delivery-failed";
 SUBSTATE=ERROR;MODIFIED=20130212T120000Z;GROUP=G1:FAILED
```

15. CalDAV Support for Task Mode

The CalDAV [RFC4791] calendar access protocol allows clients and servers to exchange iCalendar data. With the introduction of the "TASK-MODE" property in this specification, different automated task management behaviours may be delegated to the server by the Task Organizer depending upon the value of "TASK-MODE".

In order for a CalDAV client to know what task modes are available, a CalDAV server advertises a CALDAV:supported-task-mode-set WebDAV property on calendar home or calendar collections if it supports the use of the "TASK-MODE" property as described in this specification. The server can advertise a specific set of supported task modes by including one or more CALDAV:supported-task-mode XML elements within the CALDAV:supported-task-mode-set XML element. If no CALDAV:supported-task-mode XML elements are included in the WebDAV property, then clients can try any task mode, but need to be prepared for a failure when attempting to store the calendar data.

Clients MUST NOT attempt to store iCalendar data containing "TASK-MODE" elements if the CALDAV:supported-task-mode-set WebDAV property is not advertised by the server.

The server SHOULD return an HTTP 403 response with a DAV:error element containing a CALDAV:supported-task-mode XML element, if a client attempts to store iCalendar data with an "TASK-MODE" element value not supported by the server.

It is possible for a "TASK-MODE" value to be present in calendar data on the server being accessed by a client that does not support the "TASK-MODE" property. It is expected that existing clients, unaware of "TASK-MODE", will fail gracefully by ignoring the calendar property.

15.1. CALDAV:supported-task-mode-set Property

Name supported-task-mode-set

Namespace urn:ietf:params:xml:ns:caldav

Purpose Enumerates the set of supported iCalendar "TASK-MODE" element values supported by the server.

Protected This property MUST be protected and SHOULD NOT be returned by a PROPFIND allprop request (as defined in Section 14.2 of [RFC4918]).

Description See above.

Definition

```
<!ELEMENT supported-task-mode-set(supported-task-mode*)>
<!ELEMENT supported-task-mode (#PCDATA)>
<!-- PCDATA value: string - case insensitive but
uppercase preferred -->
```

Example

```
<C:supported-task-mode-set xmlns:C="urn:ietf:params:xml:ns:caldav">
  <C:supported-task-mode>AUTOMATIC-COMPLETION</C:supported-task-mode>
  <C:supported-task-mode>AUTOMATIC-FAILURE</C:supported-task-mode>
  <C:supported-task-mode>SERVER</C:supported-task-mode>
  <C:supported-task-mode>CLIENT</C:supported-task-mode>
</C:supported-task-mode-set>
```

16. Security Considerations

This specification introduces no new security considerations beyond those identified in [RFC5545].

17. IANA Considerations

17.1. Initialization of the Status registry

This specification updates [RFC5545] by adding a Status value registry to the iCalendar Elements registry and initializing it as per [RFC5545].

Name	Status	Reference
CANCELLED	Current	[RFC5545]
COMPLETED	Current	[RFC5545]
CONFIRMED	Current	[RFC5545]
DRAFT	Current	[RFC5545]
FINAL	Current	[RFC5545]
IN-PROCESS	Current	[RFC5545]
NEEDS-ACTION	Current	[RFC5545]
TENTATIVE	Current	[RFC5545]

Table 1: Initial Status Value Registry

17.2. Update of the Status registry

This specification further updates the Status registry with additional values defined in this document.

Value	Status	Reference
PENDING	Current	This Spec, Section 14.3
FAILED	Current	This Spec, Section 14.3

Table 2: Updated Status Value Registry

17.3. Sub-State value registry

The following table has been used to initialize the Sub-State registry.

Substate	Status	Reference
OK	Current	This Spec, Section 11.4
ERROR	Current	This Spec, Section 11.4
SUSPENDED	Current	This Spec, Section 11.4

Table 3: Sub-State registry

17.4. Task Mode value registry

The following table has been used to initialize the Task Mode registry.

Task Mode	Status	Reference
AUTOMATIC-COMPLETION	Current	This Spec, Section 13.2
AUTOMATIC-FAILURE	Current	This Spec, Section 13.2
CLIENT	Current	This Spec, Section 13.2
SERVER	Current	This Spec, Section 13.2

Table 4: Task Mode Value Registry

17.5. Participation Statuses registry

The following table has been used to update the Participation Statuses registry.

Value	Status	Reference
FAILED	Current	This Spec, Section 12.1

Table 5: Participation Statuses Registry

17.6. Properties registry

The following table has been used to update the Properties registry.

Property	Status	Reference
ATTENDEE	Current	This Spec, Section 14.1
COMMENT	Current	This Spec, Section 14.2
ESTIMATED_DURATION	Current	This Spec, Section 13.1

Property	Status	Reference
STATUS	Current	This Spec, Section 14.3
TASK-MODE	Current	This Spec, Section 13.2

Table 6: Updated Properties Registry

17.7. Parameters registry

The following table has been used to update the Parameters registry.

Parameter	Status	Reference
REASON	Current	This Spec, Section 11.2
MODIFIED	Current	This Spec, Section 11.3
SUBSTATE	Current	This Spec, Section 11.4

Table 7: Ipdated Parameters Registry

18. Normative References

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

[RFC4791] Daboo, C., Desruisseaux, B., and L. Dusseault, "Calendaring Extensions to WebDAV (CalDAV)", RFC 4791, IETF RFC 4791, DOI 10.17487/RFC4791, March 2007, <https://www.rfc-editor.org/info/rfc4791>.

[RFC4918] Dusseault, L., Ed., "HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV)", RFC 4918, IETF RFC 4918, DOI 10.17487/RFC4918, June 2007, <https://www.rfc-editor.org/info/rfc4918>.

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

[RFC5546] Daboo, C., Ed., "iCalendar Transport-Independent Interoperability Protocol (iTIP)", RFC 5546, IETF RFC 5546, DOI 10.17487/RFC5546, December 2009, <https://www.rfc-editor.org/info/rfc5546>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", RFC 8174, IETF RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[I-D.ietf-calext-eventpub-extensions]
Douglass, M., "Event Publishing Extensions to iCalendar", I-D.ietf-calext-eventpub-extensions, IETF I-D.ietf-calext-eventpub-extensions, March 2021.

[I-D.ietf-calext-ical-relations]
Douglass, M., "Support for iCalendar Relationships", I-D.ietf-calext-ical-relations, IETF I-D.ietf-calext-ical-relations, December 2020.

19. Informative References

[BPMN] "Business Process Model and Notation", OMG BPMN 2.0.2, January 2014, <https://www.omg.org/spec/BPMN/2.0.2/About-BPMN/>.

[I-D.york-vpoll] York, E., Daboo, C., and M. Douglass, "VPOLL: Consensus Scheduling Component for iCalendar", I-D.york-vpoll, IETF I-D.york-vpoll, February 2017.

[TARCH] "Apthorp, A., Daboo, C., Douglass, M., CalConnect, Task Architecture V1.0,", CalConnect Task Architecture V1.0.

[EDISTS] "UN Economic Commission for Europe, UN/EDIFACT, D14.A, STS STATUS, April 30, 2014,http://www.unece.org/fileadmin/DAM/trade/untdid/d14a/trsd/trsdsts.htm", UN/EDIFACT, D14.A.

[WfRP] "Russell, N., ter Hofstede, A.H.M., Edmond, T., van der Aalst,W.M.P., Workflow Resource Patterns, Eindhoven University of Technology, 2004,", WfRP.

[WSCal] "Considine, T., Douglass, M., WS-Calendar Version 1.0, OASIS, 30 July 2011,", OASIS WS-Calendar V1.0.

[WSHT] "Ings, D., Clement, L., Koenig, D., Mehta, V., Mueller, R., Rangaswamy, R., Rowley, M., Trickovic, I., Web Services - Human Task Version 1.1 (WS-HumanTask), OASIS, 17 August 2010,", OASIS WS-HT V1.1.

Appendix A. Examples of Task State Lifecycle

A.1. Simple Case Status Change

	STATUS	PARTSTAT	Action
1	-	-	Organizer draft
2	NEEDS-ACTION	NEEDS-ACTION	Organizer sends iTIP request
3		ACCEPTED	Attendee reply

	STATUS	PARTSTAT	Action
	NEEDS-ACTION		
4	PENDING	ACCEPTED	Task accepted but waiting on some "trigger" to start (e.g. another task has to finish first)
5	IN-PROCESS	IN-PROCESS	Attendee reply now working on the task
6	IN-PROCESS	COMPLETED	Attendee reply completed
7	COMPLETED	COMPLETED	Organizer changes overall state

Table 8: Example of status changes in assigning and performing a task with one attendee.

A.2. Example for multiple Attendees

Example of status changes in assigning and performing a task with two attendees (A1 and A2).

	STATUS	PARTSTAT (A1)	PARTSTAT (A2)	Action
1	-	-	-	Organizer draft.
2	NEEDS-ACTION	NEEDS-ACTION	NEEDS-ACTION	Organizer sends iTIP request.
4	NEEDS-ACTION	ACCEPTED	NEEDS-ACTION	Attendee 1 reply.
5	NEEDS-ACTION	ACCEPTED	ACCEPTED	Attendee 2 reply.
6	PENDING	ACCEPTED	ACCEPTED	Task accepted but waiting on some"trigger" to start (e.g. another task has to finish first)
7	IN-PROCESS	ACCEPTED	IN-PROCESS	Attendee 2 reply now working on the task.
8	IN-PROCESS	IN-PROCESS	IN-PROCESS	Attendee 1 reply now working on the task.
9	IN-PROCESS	COMPLETED	IN-PROCESS	Attendee 1 reply Completed (overall status still IN-PROCESS).
10	IN-PROCESS	COMPLETED	COMPLETED	Attendee 2 reply Completed
11	COMPLETED	COMPLETED	COMPLETED	Organizer changes overall state once both attendees are finished.

Table 9: Example for multiple Attendees

Note: The logic for determining the status change to the VTODO is determined by the task organizer based on the ATTENDEE status and other business logic.

A.3. Example of Failure

Example of status changes for a task that fails.

	STATUS	PARTSTAT	Action
1	-	-	Organizer draft
2	NEEDS-ACTION	NEEDS-ACTION	Organizer sends iTIP request
3	NEEDS-ACTION	ACCEPTED	Attendee reply
4	IN-PROCESS	IN-PROCESS	Attendee reply now working on the task
5	IN-PROCESS	FAILED	Attendee reply task failed
6	FAILED	FAILED	Organizer changes overall state

Table 10: Example of Failure

Appendix B. Change log

V02. 2021-05-05 MD

*Redo in asciidoc

*Change STRUCTURED-CATEGORY to CONCEPT

*Add GROUP parameter definition

V01. 2015-08-23 AA

*Highlighted use of ESTIMATED-DURATION for time planning.

*Corrected PARTSTAT example section 5.1. Changed DECLINED to FAILED.

*Replaced Task Mode AUTOMATIC-STATUS with CLIENT and SERVER modes. Also, clarified that task mode processing is only done on the organizer's copy.

*Clarified responsibility for setting MODIFIED.

*CalDAV support added.

*Updated normative references.

Appendix C. Working Notes

C.1. Advertising tasks

Use VPOLL for advertising a task to a pool of possible ATTENDEEs and then select the respondent to assign one or more assignees.

Introduce POLL-MODE:ASSIGNMENT

Need to indicate number of assignees required.

Potentially different types of response e.g. ACCEPT or DECLINE, or a weighting e.g. 0 - 100

Take into FREEBUSY discussion.

C.2. Subscribing to task updates

Stakeholders should have the ability to subscribe to categories / types of tasks on an ongoing basis. Reference calendarserver.org notifications draft

Authors' Addresses

Adrian Apthorp
DHL Express

Email: aapthorp@theiet.org

Cyrus Daboo
Apple Inc.

Email: cyrus@daboo.name

Michael Douglass
Bedework Commercial Services

Email: mdouglass@bedework.com