ONVIF™
Schedule Service Specification

Version 18.06
June 2018

CONTENTS

Contributors

ASSA ABLOY	Patrik Björling Rygert
ASSA ABLOY	Mattias Rengstedt
Axis Communications AB	Marcus Johansson
Axis Communications AB	Robert Rosengren
Axis Communications AB	Derek Wang
Axis Communications AB	Emil Selinder
Bosch	Mohane Caliaperoumal
Bosch	Dirk Schreiber
Honeywell	Uvaraj Thangarajan
Honeywell	Neelendra Bhandari
Honeywell	Mayur Salgar
Honeywell	Vinay Ghule
PACOM	Eugene Scully
Siemens AG	Lokeshwar K
Siemens AG	Suresh Raman
Siemens AG	Suresh Krishnamurthy

1 Scope

1.1 General

This specification defines the web service interface for interaction with ONVIF devices which support time management features such as schedules and special days (sometimes referred to as holidays).

Web service usage and common ONVIF functionality are outside the scope of this document. Please refer to [ONVIF Core Specification] for more information.

1.2 Conventions

The key words “shall”, “shall not”, “should”, “should not”, “may”, “need not”, “can”, “cannot” in this specification are to be interpreted as described in Annex H of [ISO/IEC Directives].

1.3 Namespaces

This document references the following namespaces:

Table 1: Referenced namespaces (with prefix)

Prefix	Namespace URI
env	http://www.w3.org/2003/05/soap-envelope
ter	http://www.onvif.org/ver10/error
xs	http://www.w3.org/2001/XMLSchema
tt	http://www.onvif.org/ver10/schema
pt	http://www.onvif.org/ver10/pacs
tns1	http://www.onvif.org/ver10/topics
tsc	http://www.onvif.org/ver10/schedule/wsdl

2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ONVIF Core Specification
<http://www.onvif.org/specs/core/ONVIF-Core-Specification.pdf>

ONVIF PACS Architecture and Design Considerations
<https://www.onvif.org/specs/wp/ONVIF-PACS-Architecture-and-Design-Considerations.pdf>

RFC 5545, *Internet Calendaring and Scheduling Core Object Specification (iCalendar), September 2009*
<https://tools.ietf.org/html/rfc5545>

RFC 5234, *Augmented BNF for Syntax Specifications: ABNF, January 2008*
<https://www.ietf.org/rfc/rfc5234.txt>

ISO/IEC Directives, *ISO/IEC Directives Part 2, Principles and rules for the structure and drafting of ISO and IEC documents, Edition 7.0, May 2016*
<http://www.iec.ch/members_experts/refdocs/iec/isoiecdir-2%7Bed7.0%7Den.pdf>

3 Terms, definitions and abbreviations

3.1 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

iCalendar	An industry standard format for exchanging scheduling and activity-recording information electronically.
Schedule	A set of time periods, e.g. working hours (weekdays from 8 AM to 6 PM). It may also include one or more special days schedules.
Special Days	A set of dates that require the regular Schedule to be overridden, e.g. holidays, half-days or working Sundays.
Special Days Schedule	A schedule that defines time periods for a Special Day List.
Time Period	A time period has a start time and an end time, e.g. 8 AM to 6 PM.
vEvent	A component in iCalendar, specifying the properties of an event.

3.2 Abbreviated terms

PACS	Physical Access Control System

4 Overview

4.1 General

The schedule service provides functions to manage schedules and special days.

Schedules can, for example, be used to define when access is granted or when a video camera should record.

The schedule service supports schedules in both local time and UTC time. Typically a schedule is in local time.

The following picture shows the main data structures involved in the schedule service:

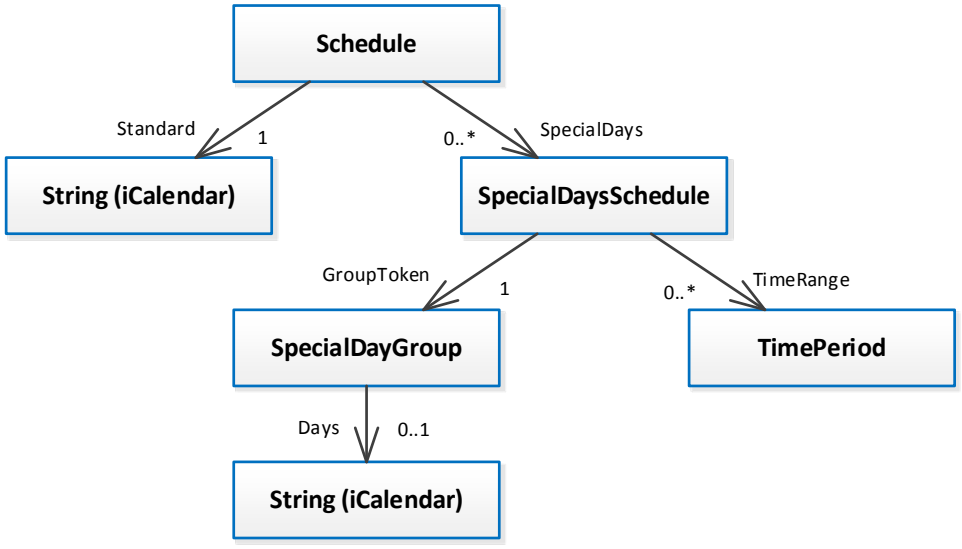

Figure 1: Main data structures in the schedule service

A group of special days may be used to define what times the facility is accessible during holidays. Another group of special days may be used to define what times the facility is accessible during half-working days.

Special day groups are typically reused by different schedules. Consider the following example:

A PACS system is shared by two sites; one in Sweden, and one in Poland. All Polish holidays are grouped in a special day group called "Polish holidays". All Swedish holidays are grouped in a special day group called "Swedish holidays". In addition Sweden have half-working days before some of the bank holidays, so another group is defined called "Swedish half-days".

A schedule called "Polish employee schedule" is created, defining access to the office for Polish employees during all days of the week, including the weekend (e.g. access on Mondays-Fridays from 7 AM to 8 PM, and access on Saturday-Sunday from 10 AM to 2 PM). The schedule is linked to the "Polish holidays" group defining the access to work during these days (e.g. no access during holidays).

Similarly, a schedule called "Swedish employee schedule" is created, defining access to the office for all Swedish employees during all days of the week, including the weekend. The schedule is linked to both the "Swedish holidays" group (e.g. no access) and the "Swedish half-days" group (e.g. access from 7 AM to 1 PM).

Schedules and special days are defined by iCalendar vEvents. iCalendar is a very flexible format where recurrence types such as yearly, monthly, weekly, daily, etc., can be used.

To briefly explain the iCalendar format, consider the following examples (only parts of the iCalendar structures are shown below):

An iCalendar event starting at 8 PM on November 25, 2014, and ending at 2 AM on November 26, 2014:

```
BEGIN:VCALENDAR
BEGIN:VEVENT
DTSTART:20141125T200000
DTEND:20141126T020000
END:VEVENT
END:VCALENDAR
```

A weekly recurring iCalendar event each Tuesday and Thursday from 2 PM to 4 PM, where the first occurrence is on November 25, 2014, and the last occurrence is on December 9, 2014 (5 occurrences):

```
BEGIN:VCALENDAR
BEGIN:VEVENT
DTSTART:20141125T140000
DTEND:20141125T160000
RRULE:FREQ=WEEKLY;COUNT=5;BYDAY=TU,TH
END:VEVENT
END:VCALENDAR
```

The following example have exactly the same effect as the previous example, but the end date of the series is specified instead of the number of occurrences:

```
BEGIN:VCALENDAR
BEGIN:VEVENT
DTSTART:20141125T140000
DTEND:20141125T160000
RRULE:FREQ=WEEKLY;UNTIL:20141209T140000;BYDAY=TU,TH
END:VEVENT
END:VCALENDAR
```

Only a certain subset of the iCalendar specification is required to be supported to get a usable service. Support for additional features within the iCalendar specification is reflected in the ServiceCapabilities.

4.2 Recurrence

4.2.1 General

When the ExtendedRecurrenceSupported capability is set to true then extended iCalendar recurrence format is supported. Special days can still be used but may not be needed.

For devices not supporting extended iCalendar recurrence, the ExtendedRecurrence-Supported capability should be set to false. The iCalendar “FREQ” and “BYDAY” keywords are used to define weekly recurring events per day in the week (see iCalendar examples above). When the ExtendedRecurrenceSupported capability is set to false, then the year of the start date of recurring events shall be set to 1970, and the occurrence count (optionally the until date) shall be omitted (i.e. forever recurring events).

The following figure shows the effect of the ExtendedRecurrenceSupported capability settings for schedule and special days respectively.

	ExtendedRecurrenceSupported	
	FALSE	**TRUE**
Schedules	Weekly recurrence	Extended recurrence
Special days	No recurrence allowed	Extended recurrence

Figure 2: Recurrence support matrix

In the following subsections, the ABNF syntax is used [RFC 5234]. The rule names to the left of the equal sign (dtstart, dtend and recur-rule-part) are redefining the rules in [RFC 5545]. The rule names to the right of the equal sign (date-month, date-mday, time-hour, time-minute,

time-second, bymolist, bywdaylist, bymodaylist, weekday and enddate) are defined according to [RFC 5545].

4.2.2 Weekly recurrence

Only the weekly iCalendar recurrence type shall be supported. Non-recurring events and other recurring types are not supported.

The device shall only accept recurring events starting with the year ‘1970’ (the month and day is needed to match the week day of the recurrence).

```
dtstart = "DTSTART" ":" "1970" date-month date-mday
          "T" time-hour time-minute time-second
```

The device shall only allow end dates no later than midnight of the date specified in the start date.

```
dtend = "DTEND" ":" "1970" date-month date-mday
        "T" time-hour time-minute time-second
```

where date-month and date-mday shall be either the same as for dtstart or one day later but with time-hour = “00”, time-minute = “00” and time-second = “00”.

Only local times are supported.

The device shall not accept an occurrence count or until date in recurring events. The following recurrence pattern is allowed:

```
recur-rule-part = "WEEKLY" [ ";" "BYDAY" "=" (weekday *("," weekday)) ]
```

4.2.3 Extended recurrence

The device shall support the following recurrence pattern:

```
recur-rule-part =
       (  ( "YEARLY" )
        / ( "YEARLY" ";" "BYMONTH" "=" bymolist )
        / ( "MONTHLY" )
        / ( "MONTHLY" ";" "BYDAY" "=" bywdaylist )
        / ( "MONTHLY" ";" "BYMONTHDAY" "=" bymodaylist )
        / ( "WEEKLY" )
        / ( "WEEKLY" ";" "BYDAY" "=" (weekday *("," weekday)) )
        / ( "DAILY" )
        / ( "HOURLY" )
        / ( "MINUTELY" )
        / ( "SECONDLY" ) )
       [ ";" "INTERVAL" "=" 1*DIGIT ]
       [ ";" "COUNT" "=" 1*DIGIT / ";" "UNTIL" "=" enddate ]
```

4.2.4 Standard schedule recurrence

Single occurrences in a series of recurring events can be deleted to create exceptions from the recurring events.

The following picture illustrates a recurring series of events. Each vertical bar represents the 24-hour day, and the green parts represent events (time periods). Events occur each Monday, but there is an exception on the fourth Monday, when no event occurs.

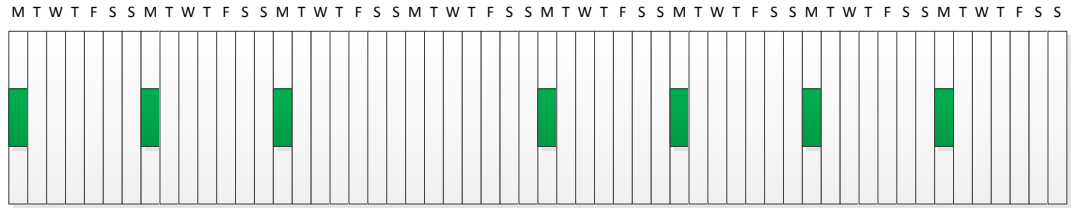

Figure 3: Recurring events with an exception

4.2.5 Special day recurrence

In contrast to exceptions, the schedule service also support special days. Special days are used to define different behavior during for instance holidays or half working days. Examples of different behavior includes video recording with a different frame rate or different access times to a facility. Special days are grouped together and given names such as “US holidays”, “UK holidays”, “Swedish half-days”, etc.

The following picture illustrates the same series of recurring events, but instead of an exception, the fourth Monday is treated as per special day with different behavior.

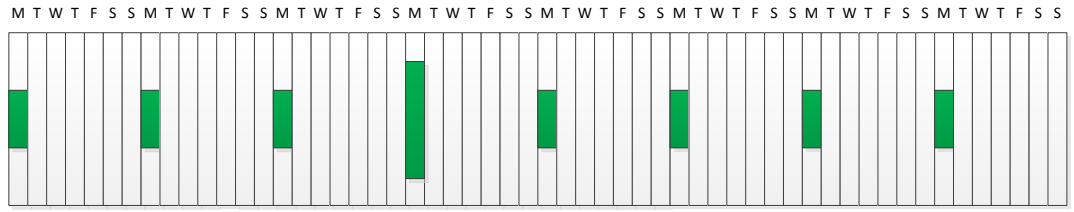

Figure 4: Recurring events with a special day

In the PACS domain, supporting different behavior during special days is essential. Most traditional access control units on the market today only handle weekly recurrence. Generally they do not support exceptions, but they usually support special days.

5 Schedules

5.1 Service capabilities

5.1.1 General

The device shall provide service capabilities in two ways:

1. With the GetServices method of Device service when IncludeCapability is true. Please refer to [ONVIF Core Specification] for more details.

2. With the GetServiceCapabilities method.

5.1.2 Data structures

5.1.2.1 ServiceCapabilities

The service capabilities reflect optional functionality of a service. The information is static and does not change during device operation. The following capabilities are available:

- **MaxLimit**

  The maximum number of entries returned by a single Get<Entity>List or Get<Entity> request. The device shall never return more than this number of entities in a single response.

- **MaxSchedules**

  Indicates the maximum number of schedules the device supports. The device shall support at least one schedule.

- **MaxTimePeriodsPerDay**

  Indicates the maximum number of time periods per day the device supports in a schedule including special days schedule. The device shall support at least one time period per day.

- **SpecialDaysSupported**

  If this capability is supported, then the device shall support special days.

- **MaxSpecialDayGroups**

  Indicates the maximum number of special day group entities the device supports. The device shall support at least one SpecialDayGroup entity if SpecialDaysSupported is set to true. This value is ignored if SpecialDaysSupported is set to false.

- **MaxDaysInSpecialDayGroup**

  Indicates the maximum number of days per SpecialDayGroup entity the device supports. The device shall support at least one day per SpecialDayGroup entity if SpecialDaysSupported is set to true. This value is ignored if SpecialDaysSupported is set to false.

- **MaxSpecialDaysSchedules**

  Indicates the maximum number of SpecialDaysSchedule entities referred by a schedule that the device supports. This value is ignored if SpecialDaysSupported is set to false.

- **ExtendedRecurrenceSupported**

  If ExtendedRecurrenceSupported = true, schedules shall support extended recurrence according to section 4.2.3.

  If ExtendedRecurrenceSupported = true and SpecialDaysSupported = true, then special days supports extended recurrence according to section 4.2.3.

  If ExtendedRecurrenceSupported = false, only weekly recurrence shall be supported for schedules (according to section 0) and no recurrence shall be supported for special days.

- **StateReportingSupported**

  If this capability is set to true, the device shall implement the GetScheduleState command, and shall notify subscribing clients whenever schedules become active or inactive.

- **ClientSuppliedTokenSupported**

  Indicates that the client is allowed to supply the token when creating schedules and special day groups. To enable the use of the commands SetSchedule and SetSpecialDayGroup, the value must be set to true.

5.1.3 GetServiceCapabilities command

This operation returns the capabilities of the schedule service.

Table 2 GetServiceCapabilities command

GetServiceCapabilities	Access Class: PRE_AUTH
Message name	**Description**
GetServiceCapabilitiesRequest	*This message shall be empty*
GetServiceCapabilitiesResponse	*This message contains:* • *"Capabilities": The capability response message contains the requested schedule service capabilities using a hierarchical XML capability structure.* tsc:ServiceCapabilities **Capabilities [1][1]**

5.2 Schedule information

5.2.1 General

A schedule is a set of time periods which may also include one or more special day's schedules.

5.2.2 Data structures

5.2.2.1 ScheduleInfo

The ScheduleInfo type represents the schedule as a physical object. The structure contains information of a specific schedule instance.

The device shall provide the following fields for each ScheduleInfo instance:

- **token**

  A service unique identifier of the schedule.

- **Name**

  A user readable name. It shall be up to 64 characters.

To provide more information, the device may include the following optional fields:

- **Description**

  User readable description for the schedule. It shall be up to 1024 characters.

5.2.2.2 Schedule

The schedule structure shall include all properties of the ScheduleInfo structure and also the standard events (iCalendar format) and a list of SpecialDaysSchedule instances.

The device shall provide the following fields for each schedule instance:

- **Standard**

  An iCalendar structure that defines a number of events. Events can be recurring or non-recurring. The events can, for instance, be used to control when a camera should record or when a facility is accessible.

  Some devices might not be able to fully support all the features of iCalendar. Setting the service capability ExtendedRecurrenceSupported to false will enable more devices to be ONVIF compliant. Is of type string (but contains an iCalendar structure).

- **SpecialDays**

  For devices that are not able to support all the features of iCalendar, supporting special days is essential. Each SpecialDaysSchedule instance defines an alternate set of time periods that overrides the regular schedule for a specified list of special days. Is of type SpecialDaysSchedule.

5.2.2.3 SpecialDaysSchedule

An override schedule that defines alternate time periods for a group of special days.

If multiple SpecialDaysSchedule structures contain conflicting time ranges for the same special day, it will result in a union of the time periods.

The picture below shows the use of SpecialDaysSchedule instances. A Schedule instance called “Employee schedule” holds an iCalendar structure defining the regular access, and also holds a number of SpecialDaysSchedule instances. Each SpecialDaysSchedule holds a token to a SpecialDayGroup and zero or more TimePeriod instances.

In this example, the first SpecialDaysSchedule instance holds the token to the “Holidays” special day group, but holds no time periods. The result is that no access is granted on the special days listed in the “Holidays” special day group.

The second SpecialDaysSchedule instance holds the token to the “Half-days” special day group, and also holds one time period “7 AM to 1 PM”, resulting in access granted during those hours for special days listed in the “Half-days” special day group.

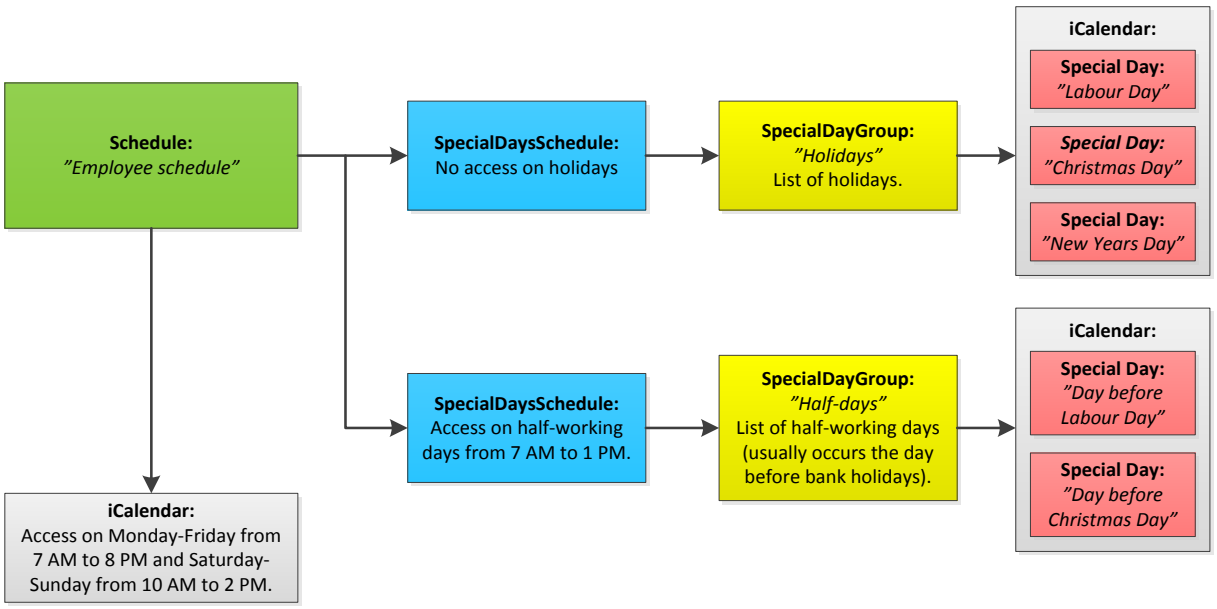

Figure 5: SpecialDaysSchedule example

If a special day is defined as part of a day (requires ExtendedRecurrenceSupported = true), then access shall only be granted when the time range and the part of the special day overlaps.

Consider the example depicted in the figure below. Assume a standard schedule giving access on weekdays 6 AM to 6 PM (marked in green in Figure 6), but not on weekends. Then assume a special day starting on Friday, June 10, at 2 PM and ending on Saturday, June 11, at 10 AM (marked in yellow in Figure 6). If the special days' schedule defines a time range of 8 AM to 4 PM (marked by dashed red lines in Figure 6), then access will be granted from 6 AM to 4 PM on Friday, June 10, and from 8 AM to 10 AM on Saturday, June 11. See detailed example in section A.8.

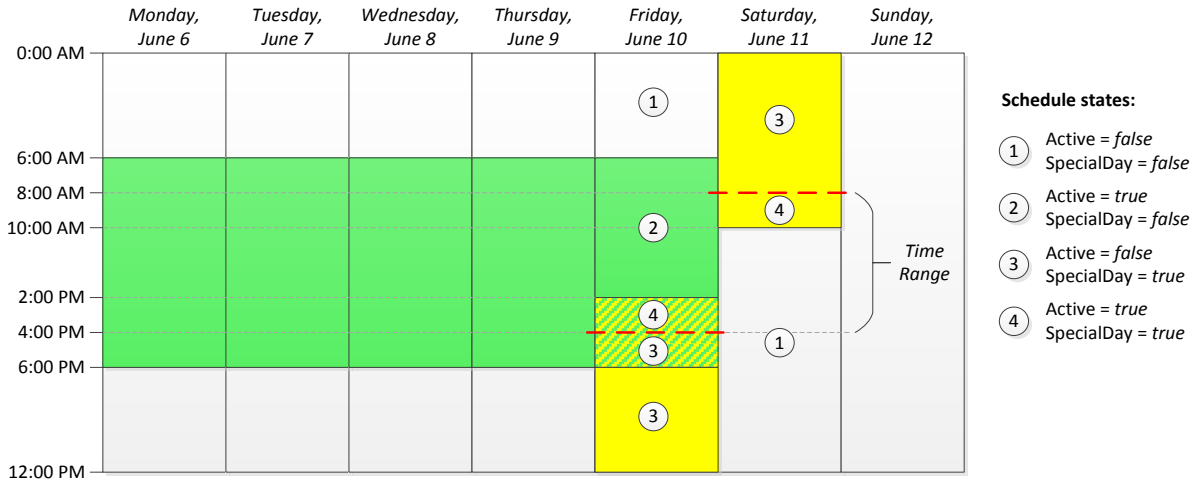

Figure 6: Example of special day with time part

The device shall provide the following fields for each SpecialDaysSchedule instance:

- **GroupToken**

  Indicates the list of special days in a schedule.

- **TimeRange**

  Indicates the alternate time periods for the list of special days (overrides the regular schedule). For example, the regular schedule indicates that it is active from 8 AM to 5 PM on Mondays. However, this particular Monday is a special day, and the alternate time periods state that the schedule is active from 9 AM to 11 AM and 1 PM to 4 PM.

  If no time periods are defined, then no access is allowed. Is of type TimePeriod.

5.2.2.4 TimePeriod

A time period defines a start and end time. For full day access, the start time = ”00:00:00” with no defined end time. For a time period with no end time, the schedule runs until midnight. If an end time is specified, it must always be greater than the start time, otherwise an InvalidArgVal fault shall be generated by the device.

The device shall provide the following fields for each TimePeriod instance:

- **From**

  Indicates the start time.

To provide more information, the device may include the following optional fields:

- **Until**

  Indicates the end time. Is optional, if omitted, the period ends at midnight. The end time is exclusive, meaning that that exact moment in time is not part of the period. To determine if a moment in time (t) is part of a time period, the formula StartTime ≤ t < EndTime is used.

5.2.3 GetScheduleInfo command

This operation requests a list of ScheduleInfo items matching the given tokens.

The device shall ignore tokens it cannot resolve and shall return an empty list if there are no items matching the specified tokens. The device shall not return a fault in this case.

If the number of requested items is greater than MaxLimit, a TooManyItems fault shall be returned.

Table 3 GetScheduleInfo command

GetScheduleInfo	Access Class: READ_SYSTEM
Message name	**Description**
GetScheduleInfoRequest	*This message contains:* • *"Token": Tokens of ScheduleInfo items to get.* pt:ReferenceToken **Token [1][unbounded]**
GetScheduleInfoResponse	*This message contains:* • *"ScheduleInfo": List of ScheduleInfo items.* tsc:ScheduleInfo **ScheduleInfo [0][unbounded]**
Fault codes	**Description**
env:Sender ter:InvalidArgs ter:TooManyItems	*Too many items were requested, see MaxLimit capability.*

5.2.4 GetScheduleInfoList command

This operation requests a list of all ScheduleInfo items provided by the device.

A call to this method shall return a StartReference when not all data is returned and more data is available. The reference shall be valid for retrieving the next set of data. Please refer to section 4.8.3 in [ONVIF PACS Architecture and Design Considerations] for more details.

The number of items returned shall not be greater than the Limit parameter.

Table 4 GetScheduleInfoList command

GetScheduleInfoList	Access Class: READ_SYSTEM
Message name	**Description**
GetScheduleInfoListRequest	*This message contains:* • *"Limit": Maximum number of entries to return. If not specified, less than one or higher than what the device supports, the number of items is determined by the device.* • *"StartReference": Start returning entries from this start reference. If not specified, entries shall start from the beginning of the dataset.* xs:int **Limit [0][1]** xs:string **StartReference [0][1]**
GetScheduleInfoListResponse	*This message contains:* • *"NextStartReference": StartReference to use in next call to get the following items. If absent, no more items to get.* • *"ScheduleInfo": List of ScheduleInfo items.* xs:string **NextStartReference [0][1]** tsc:ScheduleInfo **ScheduleInfo [0][unbounded]**
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:InvalidStartReference	*StartReference is invalid or has timed out. Client needs to start fetching from the beginning.*

5.2.5 GetSchedules command

This operation requests a list of Schedule items matching the given tokens.

The device shall ignore tokens it cannot resolve and shall return an empty list if there are no items matching the specified tokens. The device shall not return a fault in this case.

If the number of requested items is greater than MaxLimit, a TooManyItems fault shall be returned.

Table 5 GetSchedules command

GetSchedules	Access Class: READ_SYSTEM
Message name	**Description**
GetScheduleRequest	*This message contains:* • "*Token": Tokens of Schedule items to get* pt:ReferenceToken **Token [1][unbounded]**
GetScheduleResponse	*This message contains:* • *"Schedule": List of schedule items.* tsc:Schedule **Schedule [0][unbounded]**
Fault codes	**Description**
env:Sender ter:InvalidArgs ter:TooManyItems	*Too many items were requested, see MaxLimit capability.*

5.2.6 GetScheduleList command

This operation requests a list of all Schedule items provided by the device.

A call to this method shall return a StartReference when not all data is returned and more data is available. The reference shall be valid for retrieving the next set of data. Please refer to section 4.8.3 in [ONVIF PACS Architecture and Design Considerations] for more details.

The number of items returned shall not be greater than the Limit parameter.

Table 6 GetScheduleList command

GetScheduleList	Access Class: READ_SYSTEM
Message name	**Description**
GetScheduleListRequest	*This message contains:* • *"Limit": Maximum number of entries to return. If not specified, less than one or higher than what the device supports, the number of items is determined by the device.* • *"StartReference": Start returning entries from this start reference. If not specified, entries shall start from the beginning of the dataset.* xs:int **Limit [0][1]** xs:string **StartReference [0][1]**
GetScheduleListResponse	*This message contains:* • *"NextStartReference": StartReference to use in next call to get the following items. If absent, no more items to get.* • *"Schedule": List of Schedule items.* xs:string **NextStartReference [0][1]** tsc:Schedule **Schedule [0][unbounded]**
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:InvalidStartReference	*StartReference is invalid or has timed out. Client needs to start fetching from the beginning.*

5.2.7 CreateSchedule command

This operation creates the specified schedule in the device.

The token field of the Schedule structure shall be empty and the device shall allocate a token for the schedule. The allocated token shall be returned in the response.

If the client sends any value in the token field, the device shall return InvalidArgVal as a generic fault code.

Table 7 CreateSchedule command

CreateSchedule	Access Class: WRITE_SYSTEM
Message name	**Description**
CreateScheduleRequest	*This message contains:* • *"Schedule": The Schedule to create* tsc:Schedule **Schedule [1][1]**
CreateScheduleResponse	*This message contains:* • *"Token": The token of created Schedule* pt:ReferenceToken **Token [1][1]**
Fault codes	**Description**
env:Receiver ter:CapabilityViolated ter:MaxSchedules	*There is not enough space to add new schedule, see MaxSchedules capability*
env:Sender ter: CapabilityViolated ter:MaxSpecialDaysSchedules	*There are too many SpecialDaysSchedule entities referred in this schedule, see MaxSpecialDaysSchedules capability.*
env:Sender ter: CapabilityViolated ter:MaxTimePeriodsPerDay	*There are too many time periods in a day schedule, see MaxTimePeriodsPerDay capability.*
env:Sender ter:InvalidArgVal ter:ReferenceNotFound	*A referred entity token is not found (some devices may not validate referred entities).*

5.2.8 SetSchedule command

This method is used to synchronize a schedule in a client with the device.

If a schedule with the specified token does not exist in the device, the schedule is created. If a schedule with the specified token exists, then the schedule is modified.

A call to this method takes a Schedule structure as input parameter. The token field of the Schedule structure must not be empty.

A device that signals support for the ClientSuppliedTokenSupported capability shall implement this command.

If no token was specified in the request, the device shall return InvalidArgs as a generic fault code.

Table 8 SetSchedule command

SetSchedule	Access Class: WRITE_SYSTEM
Message name	**Description**
SetScheduleRequest	*This message contains:* • *"Schedule": The Schedule item to create or modify* tsc:Schedule **Schedule [1][1]**
SetScheduleResponse	*This message shall be empty*
Fault codes	**Description**
env:Receiver ter:CapabilityViolated ter:ClientSuppliedTokenSupported	*The device does not support that the client supplies the token*
env:Receiver ter:CapabilityViolated ter:MaxSchedules	*There is not enough space to add new schedule, see MaxSchedules capability*
env:Sender ter:CapabilityViolated ter:MaxSpecialDaysSchedules	*There are too many SpecialDaysSchedule entities referred in this schedule, see MaxSpecialDaysSchedules capability.*
env:Sender ter:CapabilityViolated ter:MaxTimePeriodsPerDay	*There are too many time periods in a day schedule, see MaxTimePeriodsPerDay capability.*
env:Sender ter:InvalidArgVal ter:ReferenceNotFound	*A referred entity token is not found (some devices may not validate referred entities).*

5.2.9 ModifySchedule command

This operation modifies the specified schedule.

The token of the schedule to modify is specified in the token field of the Schedule structure and shall not be empty. All other fields in the structure shall overwrite the fields in the specified schedule.

If no token was specified in the request, the device shall return InvalidArgs as a generic fault code.

Table 9 ModifySchedule command

ModifySchedule	Access Class: WRITE_SYSTEM
Message name	**Description**
ModifyScheduleRequest	*This message contains:* • *"Schedule": The Schedule to modify/update* tsc:Schedule **Schedule [1][1]**
ModifyScheduleResponse	*This message shall be empty*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token not found.*
env:Sender ter:CapabilityViolated ter:MaxSpecialDaysSchedules	*There are too many SpecialDaysSchedule entities referred in this schedule, see MaxSpecialDaysSchedules capability.*
env:Sender ter: CapabilityViolated ter:MaxTimePeriodsPerDay	*There are too many time periods in a day schedule, see MaxTimePeriodsPerDay capability.*
env:Sender ter:InvalidArgVal ter:ReferenceNotFound	*A referred entity token is not found (some devices may not validate referred entities).*

5.2.10 DeleteSchedule command

This operation will delete the specified schedule.

If it is associated with one or more entities some devices may not be able to delete the schedule, and consequently a ReferenceInUse fault shall be generated.

If no token was specified in the request, the device shall return InvalidArgs as a generic fault code.

Table 10 DeleteSchedule command

DeleteSchedule	Access Class: WRITE_SYSTEM
Message name	**Description**
DeleteScheduleRequest	*This message contains:* • *"Token": The token of the schedule to delete.* pt:ReferenceToken **Token [1][1]**
DeleteScheduleResponse	*This message shall be empty*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*Schedule token is not found.*
env:Sender ter:InvalidArgVal ter:ReferenceInUse	*Failed to delete, Schedule token is in use*

5.3 Special day group information

5.3.1 General

A special day group are days (or parts of days) that require the regular schedule to be overridden with an alternate schedule. For example holidays, half-days, working Sundays, etc.

5.3.2 Data structures

5.3.2.1 SpecialDayGroupInfo

The SpecialDayGroupInfo structure contains the basic information about the special days list.

The device shall provide the following fields for each SpecialDayGroupInfo instance:

- **token**
  A service-unique identifier of the special day group.
- **Name**
  User readable name. It shall be up to 64 characters.

To provide more information, the device may include the following optional fields:

- **Description**
  User readable description for the special days. It shall be up to 1024 characters.

5.3.2.2 SpecialDayGroup

The special day group structure shall include all properties of the SpecialDayGroupInfo structure and also a set of special days. A special day group are days (or parts of days) that

require the regular schedule to be overridden with an alternate schedule. For example holidays, half-days, working Sundays, etc.

The device shall provide the following fields for each SpecialDayGroup instance:

- **Days**

  An iCalendar structure that contains a group of special days. Is of type string (containing an iCalendar structure).

5.3.3 GetSpecialDayGroupInfo command

This operation requests a list of SpecialDayGroupInfo items matching the given tokens.

The device shall ignore tokens it cannot resolve and shall return an empty list if there are no items matching specified tokens. The device shall not return a fault in this case.

If the number of requested items is greater than MaxLimit, a TooManyItems fault shall be returned.

Table 11 GetSpecialDayGroupInfo command

GetSpecialDayGroupInfo	Access Class: READ_SYSTEM
Message name	**Description**
GetSpecialDayGroupInfoRequest	*This message contains:* • "*Token": Tokens of SpecialDayGroupInfo items to get.* pt:ReferenceToken **Token [1][unbounded]**
GetSpecialDayGroupInfoResponse	*This message contains:* • *"SpecialDayGroupInfo": List of SpecialDayGroupInfo items.* tsc:SpecialDayGroupInfo **SpecialDayGroupInfo [0][unbounded]**
Fault codes	**Description**
env:Sender ter:InvalidArgs ter:TooManyItems	*Too many items were requested, see MaxLimit capability.*

5.3.4 GetSpecialDayGroupInfoList command

This operation requests a list of all SpecialDayGroupInfo items provided by the device.

A call to this method shall return a StartReference when not all data is returned and more data is available. The reference shall be valid for retrieving the next set of data. Please refer to section 4.8.3 in [ONVIF PACS Architecture and Design Considerations] for more details.

The number of items returned shall not be greater than the Limit parameter.

Table 12 GetSpecialDayGroupInfoList command

GetSpecialDayGroupInfoList	Access Class: READ_SYSTEM
Message name	**Description**
GetSpecialDayGroupInfoListRequest	*This message contains:* • *"Limit": Maximum number of entries to return. If not specified, less than one or higher than what the device supports, the number of items is determined by the device.* • *"StartReference": Start returning entries from this start reference. If not specified, entries shall start from the beginning of the dataset.* xs:int **Limit [0][1]** xs:string **StartReference [0][1]**
GetSpecialDayGroupInfoListResponse	*This message contains:* • *"NextStartReference": StartReference to use in next call to get the following items. If absent, no more items to get.* • *"SpecialDayGroupInfo": List of SpecialDayGroupInfo items.* xs:string **NextStartReference [0][1]** tsc:SpecialDayGroupInfo **SpecialDayGroupInfo [0][unbounded]**
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:InvalidStartReference	*StartReference is invalid or has timed out. Client needs to start fetching from the beginning.*

5.3.5 GetSpecialDayGroups command

This operation requests a list of SpecialDayGroup items matching the given tokens.

The device shall ignore tokens it cannot resolve and shall return an empty list if there are no items matching the specified tokens. The device shall not return a fault in this case.

If the number of requested items is greater than MaxLimit, a TooManyItems fault shall be returned.

Table 13 GetSpecialDayGroups command

GetSpecialDayGroups	Access Class: READ_SYSTEM
Message name	**Description**
GetSpecialDayGroupsRequest	*This message contains:* • "*Token": Tokens of the SpecialDayGroup items to get.* pt:ReferenceToken **Token [1][unbounded]**
GetSpecialDayGroupsResponse	*This message contains:* • *"SpecialDayGroup": List of SpecialDayGroup items.* tsc:SpecialDayGroup **SpecialDayGroup [0][unbounded]**
Fault codes	**Description**
env:Sender ter:InvalidArgs ter:TooManyItems	*Too many items were requested, see MaxLimit capability.*

5.3.6 GetSpecialDayGroupList command

This operation requests a list of all SpecialDayGroupList items provided by the device.

A call to this method shall return a StartReference when not all data is returned and more data is available. The reference shall be valid for retrieving the next set of data. Please refer to section 4.8.3 in [ONVIF PACS Architecture and Design Considerations] for more details.

The number of items returned shall not be greater than the Limit parameter.

Table 14 GetSpecialDayGroupList command

GetSpecialDayGroupList	Access Class: READ_SYSTEM
Message name	**Description**
GetSpecialDayGroupListRequest	*This message contains:* • *"Limit": Maximum number of entries to return. If not specified, less than one or higher than what the device supports, the number of items is determined by the device.* • *"StartReference": Start returning entries from this start reference. If not specified, entries shall start from the beginning of the dataset.* xs:int **Limit [0][1]** xs:string **StartReference [0][1]**
GetSpecialDayGroupListResponse	*This message contains:* • *"NextStartReference": StartReference to use in next call to get the following items. If absent, no more items to get.* • *"SpecialDayGroup": List of SpecialDayGroup items.* xs:string **NextStartReference [0][1]** tsc:SpecialDayGroup **SpecialDayGroup [0][unbounded]**
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:InvalidStartReference	*StartReference is invalid or has timed out. Client needs to start fetching from the beginning.*

5.3.7 CreateSpecialDayGroup command

This operation creates the specified special day group in the device.

The token field of the SpecialDayGroup structure shall be empty and the device shall allocate a token for the special day group. The allocated token shall be returned in the response.

If the client sends any value in the token field, the device shall return InvalidArgVal as a generic fault code.

Table 15 CreateSpecialDayGroup command

CreateSpecialDayGroup	Access Class: WRITE_SYSTEM
Message name	**Description**
CreateSpecialDayGroupRequest	*This message contains:* • *"SpecialDayGroup": The special day group to create.* tsc:SpecialDayGroup **SpecialDayGroup [1][1]**
CreateSpecialDayGroupResponse	*This message contains:* • *"Token": The token of created special day group.* pt:ReferenceToken **Token [1][1]**
Fault codes	**Description**
env:Receiver ter:CapabilityViolated ter:MaxSpecialDayGroups	*There is not enough space to add new SpecialDayGroup items, see the MaxSpecialDayGroups capability*
env:Sender ter:CapabilityViolated ter:MaxDaysInSpecialDayGroup	*There are too many special days in a SpecialDayGroup, see MaxDaysInSpecialDayGroup capability.*

5.3.8 SetSpecialDayGroup command

This method is used to synchronize a special day group in a client with the device.

If a special day group with the specified token does not exist in the device, the special day group is created. If a special day group with the specified token exists, then the special day group is modified.

A call to this method takes a special day group structure as input parameter. The token field of the SpecialDayGroup structure shall not be empty.

A device that signals support for the ClientSuppliedTokenSupported capability shall implement this command.

Table 16 SetSpecialDayGroup command

SetSpecialDayGroup	Access Class: WRITE_SYSTEM
Message name	**Description**
SetSpecialDayGroupRequest	*This message contains:* • *"SpecialDayGroup": The SpecialDayGroup item to create or modify* tsc:SpecialDayGroup **SpecialDayGroup [1][1]**
SetSpecialDayGroupResponse	*This message shall be empty*
Fault codes	**Description**
env:Receiver ter:CapabilityViolated ter:ClientSuppliedTokenSupported	*The device does not support that the client supplies the token*
env:Receiver ter:CapabilityViolated ter:MaxSpecialDayGroups	*There is not enough space to add new SpecialDayGroup items, see the MaxSpecialDayGroups capability*
env:Sender ter:CapabilityViolated ter:MaxDaysInSpecialDayGroup	*There are too many special days in a SpecialDayGroup, see MaxDaysInSpecialDayGroup capability.*
env:Sender ter:InvalidArgVal ter:MissingToken	*The token of the special day group item must be specified.*

5.3.9 ModifySpecialDayGroup command

This operation modifies the specified special day group.

The token of the special day group to modify is specified in the token field of the SpecialDayGroup structure and shall not be empty. All other fields in the structure shall overwrite the fields in the specified special day group.

Table 17 ModifySpecialDayGroup command

ModifySpecialDayGroup	Access Class: WRITE_SYSTEM
Message name	**Description**
ModifSpecialDayGroupRequest	*This message contains:* • *"SpecialDayGroup": The SpecialDayGroup item to modify/update.* tsc:SpecialDayGroup **SpecialDayGroup [1][1]**
ModifySpecialDayGroupResponse	*This message shall be empty*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token not found.*
env:Sender ter: CapabilityViolated ter:MaxDaysInSpecialDayGroup	*There are too many special days in a SpecialDayGroup, see MaxDaysInSpecialDayGroup capability.*

5.3.10 DeleteSpecialDayGroup command

This method deletes the specified special day group.

If it is associated with one or more schedules some devices may not be able to delete the special day group, and consequently a ReferenceInUse fault must be generated.

Table 18 DeleteSpecialDayGroup command

DeleteSpecialDayGroup	Access Class: WRITE_SYSTEM
Message name	**Description**
DeleteSpecialDayGroupRequest	*This message contains:* • *"Token": The token of the SpecialDayGroup item to delete.* pt:ReferenceToken **Token [1][1]**
DeleteSpecialDayGroupResponse	*This message shall be empty*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*Special day group token is not found.*
env:Sender ter:InvalidArgVal ter:ReferenceInUse	*Failed to delete, Special day group token is in use*

5.4 Schedule status

5.4.1 Data structures

5.4.1.1 ScheduleState

The ScheduleState contains state information for a schedule. The two fields below forms four different states that a schedule can have.

The device shall provide the following fields for each ScheduleState instance:

- **Active**

  Indicates that the current time is within the boundaries of the schedule or its special days schedules' time periods. For example, if this schedule is being used for triggering automatic recording on a video source, the Active flag will be true when the schedule-based recording is supposed to record.

To provide more information, the device may include the following optional field:

- **SpecialDay**

  Indicates that the current time is within the boundaries of its special days schedules' time periods. For example, if this schedule is being used for recording at a lower frame rate on a video source during special days, the SpecialDay flag will be true. If special days are not supported by the device, this field may be omitted and interpreted as false by the client.

Example:

Consider a schedule defining a standard schedule that is active from 9 AM to 5 PM during week days. There are two different special day groups; one defining half working days, and one defining bank holidays. The special days schedule defines a time period from 9 AM to 1 PM on half working days.

Figure 7 depicts when the four different states occur:

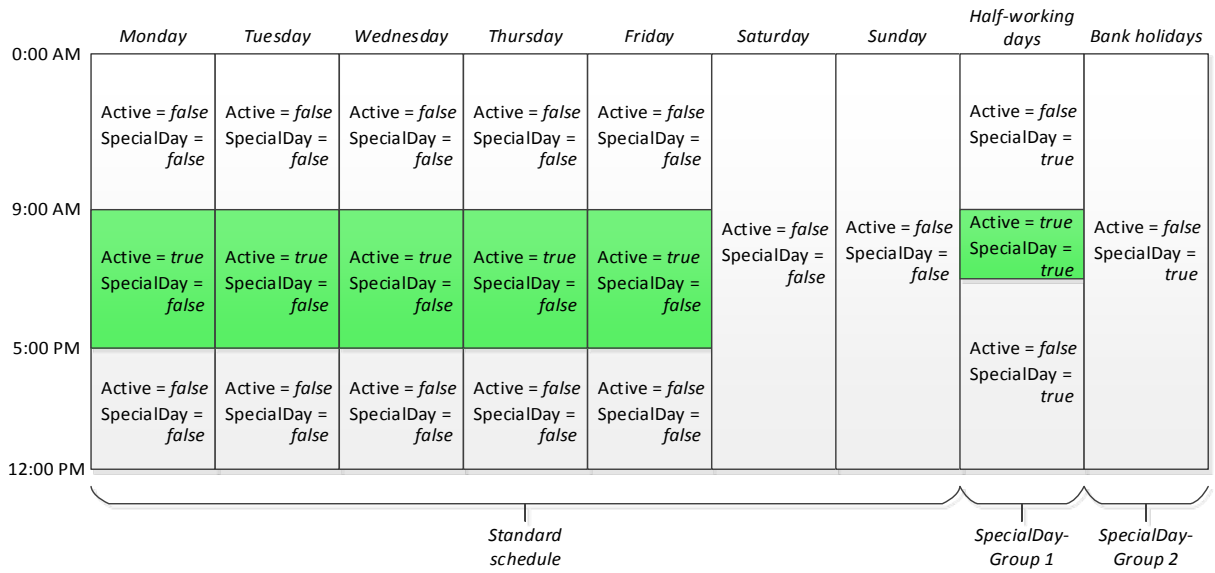

Figure 7: Schedule states

5.4.2 GetScheduleState command

This operation requests the ScheduleState for the schedule instance specified by the given token.

Table 19 GetScheduleState command

GetScheduleState	Access Class: READ_SYSTEM_SENSITIVE
Message name	**Description**
GetScheduleStateRequest	*This message contains:* • *"Token": Token of schedule instance to get ScheduleState.* pt:ReferenceToken **Token [1][1]**
GetScheduleStateResponse	*This message contains:* • *"ScheduleState": ScheduleState item.* tsc:ScheduleState **ScheduleState [1][1]**
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*Schedule token is not found*

6 Notification topics

6.1 General

This section defines notification topics specific to the schedule service.

6.2 Event overview (informative)

The schedule service specifies events when schedules or its special days are active and when schedules or special days are changed.

The main topic for status changes are:

- tns1:Schedule/State/Active

The main topics for configuration change notifications are:

- tns1:Configuration/Schedule/Changed
- tns1:Configuration/Schedule/Removed
- tns1:Configuration/SpecialDays/Changed
- tns1:Configuration/SpecialDays/Removed

6.3 Status changes

6.3.1 General

The device shall provide the status change events to inform subscribed clients when schedule entity status is changed. The device shall use the topics defined in this section associated with the respective message description.

6.3.2 Schedule

If the StateReportingSupported capability is set to true then the service/device shall be capable of generating the following event whenever a schedule or its special days becomes active or inactive based on the device time. It's a property event that indicates if the schedule is active or not.

```
Topic: tns1:Schedule/State/Active

<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItemDescription Name="ScheduleToken"
                              Type="pt:ReferenceToken"/>
    <tt:SimpleItemDescription Name="Name"
                              Type="xs:string"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="Active"
                              Type="xs:boolean"/>
    <tt:SimpleItemDescription Name="SpecialDay"
                              Type="xs:boolean"/>
  </tt:Data>
</tt:MessageDescription>
```

6.4 Configuration changes

6.4.1 General

Whenever configuration data has been changed, added or been removed an ONVIF compliant device shall provide these events to inform subscribed clients.

6.4.2 Schedule

Whenever the configuration data for a schedule is changed (including SpecialDaysSchedule) or if a schedule is added, the device shall provide the following event:

```
Topic: tns1:Configuration/Schedule/Changed

<tt:MessageDescription IsProperty=”false”>
  <tt:Source>
    <tt:SimpleItemDescription Name=”ScheduleToken”
                              Type=”pt:ReferenceToken”/>
  </tt:Source>
</tt: MessageDescription>
```

Whenever a schedule is removed, the device shall provide the following event:

```
Topic: tns1:Configuration/Schedule/Removed

<tt:MessageDescription IsProperty=”false”>
  <tt:Source>
    <tt:SimpleItemDescription Name=”ScheduleToken”
                              Type=”pt:ReferenceToken”/>
  </tt:Source>
</tt:MessageDescription>
```

6.4.3 Special day group

Whenever the configuration data for a SpecialDayGroup item is changed or added, the device shall provide the following event:

```
Topic: tns1:Configuration/SpecialDays/Changed

<tt:MessageDescription IsProperty=”false”>
  <tt:Source>
    <tt:SimpleItemDescription Name=”SpecialDaysToken”
                              Type=”pt:ReferenceToken”/>
  </tt:Source>
</tt:MessageDescription>
```

Whenever a SpecialDayGroup item is removed, the device shall provide the following event:

```
Topic: tns1:Configuration/SpecialDays/Removed

<tt:MessageDescription IsProperty=”false”>
  <tt:Source>
    <tt:SimpleItemDescription Name=”SpecialDaysToken”
                              Type=”pt:ReferenceToken”/>
  </tt:Source>
</tt:MessageDescription>
```

Annex A - Examples

Note that all iCalendar structures in the examples below are not complete. Only the parts that are explanatory are included.

*A.1 Access 24*7 for admin staff*

In the following example, the values of SpecialDaysSupported and ExtendedRecurrence-Supported are irrelevant.

```
<Schedule token=″402″>
       <Name>Access 24*7</Name>
       <Standard>
              BEGIN:VCALENDAR
              BEGIN:VEVENT
              SUMMARY:Access 24*7
              DTSTART:19700101T000000
              DTEND:19700102T000000
              RRULE:FREQ=WEEKLY;BYDAY=MO,TU,WE,TH,FR,SA,SU
              END:VEVENT
              END:VCALENDAR
       </Standard>
</Schedule>
```

A.2 Access on Monday and Wednesday from 6 AM to 8 PM for cleaning staff

In the following example, the values of SpecialDaysSupported and ExtendedRecurrence-Supported are irrelevant.

```
<Schedule token=″402″>
       <Name>Access on Monday and Wednesday from 6 AM to 8 PM</Name>
       <Standard>
              BEGIN:VCALENDAR
              BEGIN:VEVENT
              SUMMARY:Access on Monday and Wednesday from 6 AM to 8 PM
              DTSTART:19700105T060000
              DTEND:19700105T200000
              RRULE:FREQ=WEEKLY;BYDAY=MO,WE
              END:VEVENT
              END:VCALENDAR
       </Standard>
</Schedule>
```

A.3 Access from Friday 6 PM to Monday 7 AM for maintenance staff

This example can be realized in two different ways depending on the capabilities of the schedule service exposed by the device.

Case A

The following example is valid for ExtendedRecurrenceSupported=true. The value of Special-DaysSupported is irrelevant.

Note that if ExtendedRecurrenceSupported=false for the example below, the device will throw a fault because the year in DTSTART is not ‘1970’.

```
<Schedule token=″402″>
       <Name>Access from Friday 6 PM to Monday 7 AM for maintenance staff</Name>
       <Standard>
              BEGIN:VCALENDAR
              BEGIN:VEVENT
              SUMMARY:Access from Friday 6 PM to Monday 7 AM for maintenance staff
              DTSTART:20140523T180000
              DTEND:20140526T070000
              RRULE:FREQ=WEEKLY;BYDAY=FR
              END:VEVENT
              END:VCALENDAR
       </Standard>
</Schedule>
```

Case B

The following example is valid for SpecialDaysSupported=false. The value of Extended-RecurrenceSupported is irrelevant.

```
<Schedule token=”402”>
       <Name>Access from Friday 6 PM to Monday 7 AM for maintenance staff</Name>
       <Standard>
              BEGIN:VCALENDAR
              BEGIN:VEVENT
              SUMMARY:Access from Friday 6 PM to Friday midnight
              DTSTART:19700102T180000
              DTEND:19700103T000000
              RRULE:FREQ=WEEKLY;BYDAY=FR
              END:VEVENT
              BEGIN:VEVENT
              SUMMARY:Access for all Saturdays and all Sundays
              DTSTART:19700103T000000
              DTEND:19700104T000000
              RRULE:FREQ=WEEKLY;BYDAY=SA,SU
              END:VEVENT
              BEGIN:VEVENT
              SUMMARY:Access from Sunday midnight to Monday 7 AM
              DTSTART:19700105T000000
              DTEND:19700105T070000
              RRULE:FREQ=WEEKLY;BYDAY=MO
              END:VEVENT
              END:VCALENDAR
       </Standard>
</Schedule>
```

A.4 Access on Weekdays from 8 AM to 5 PM for employees

In the following example, the values of SpecialDaysSupported and ExtendedRecurrence-Supported are irrelevant.

```
<Schedule token=”402”>
       <Name>Access on weekdays from 8 AM to 5 PM for employees</Name>
       <Standard>
              BEGIN:VCALENDAR
              BEGIN:VEVENT
              SUMMARY:Access on weekdays from 8 AM to 5 PM for employees
              DTSTART:19700101T080000
              DTEND: 19700101T170000
              RRULE:FREQ=WEEKLY;BYDAY=MO,TU,WE,TH,FR
              END:VEVENT
              END:VCALENDAR
       </Standard>
</Schedule>
```

A.5 Access from January 15, 2014, to January 14, 2015, from 9 AM to 6 PM

This example can be realized in two different ways depending on the capabilities of the schedule service exposed by the device.

Case A

The following example is valid for ExtendedRecurrenceSupported=true. The value of Special-DaysSupported is irrelevant.

```
<Schedule token=”402”>
       <Name>Access from Jan 15, 2014, to Jan 14, 2015, from 9 AM to 6 PM</Name>
       <Standard>
              BEGIN:VCALENDAR
              BEGIN:VEVENT
              SUMMARY:Access from Jan 15, 2014, to Jan 14, 2015, from 9 AM to 6 PM
              DTSTART:20140115T090000
              DTEND:20150114T180000
              RRULE:FREQ=DAILY
              END:VEVENT
              END:VCALENDAR
       </Standard>
</Schedule>
```

Case B

The following example is valid for SpecialDaysSupported=false and ExtendedRecurrence-Supported=false.

Note: This scenario cannot be directly implemented in the device using only the schedule if ExtendedRecurrenceSupported capability is not supported. PACS uses start and end dates in access policies which refer schedules. Alternately, the client or the host would need to break the schedule into weekly schedule (whatever recurrence the device supports) and download those schedules to the device whenever the schedule changes (January 15, 2014 and January 14, 2015).

A.6 Special Days example 1

An employee has access defined as per the schedule, refer example A.4. However the site admin wants to restrict access to employees on Christmas day (Thursday) as it is a national holiday. The site admin has a special days entity called ‘National Holidays’ with December 25 added to it. The site admin modifies the schedule to add a special days schedule for the special days entity ‘National Holidays’ to override the access and disable employee from accessing the facility.

The following example is valid for SpecialDaysSupported=true. The value of Extended-RecurrenceSupported is irrelevant.

```
<SpecialDayGroup token=”8765”>
	<Name>National Holidays</Name>
	<Days>
		BEGIN:VCALENDAR
		PRODID:VERSION:2.0
		BEGIN:VEVENT
		SUMMARY:Christmas day
		DTSTART:20141225T000000
		DTEND:20141226T000000
		UID:Holiday@ONVIF.com
		END:VEVENT
		END:VCALENDAR
	</Days>
<SpecialDayGroup>

<Schedule token=”402”>
	<Name>Access on Monday and Wednesday from 8 AM to 5 PM. No access on Christmas
day</Name>
	<Standard>
		BEGIN:VCALENDAR
		PRODID:VERSION:2.0
		BEGIN:VEVENT
		SUMMARY:Access on Monday and Wednesday from 8 AM to 5 PM
		DTSTART:19700105T080000
		DTEND:19700105T170000
		RRULE:FREQ=WEEKLY;BYDAY=MO,WE
		UID:Event@ONVIF.com
		END:VEVENT
		END:VCALENDAR
	</Standard>
	<SpecialDaysSchedule>
		<GroupToken>8765</GroupToken>
		<TimeRange></TimeRange>
		<!-- No time period indicates no access on referred special days -->
	</SpecialDaysSchedule>
</Schedule>
```

A.7 Special Days example 2

A janitor has access defined as per schedule refer example A.4, however the site admin wants to allow access to cleaning staff on Christmas day (Thursday) as it is a national holiday. The site admin has a special days entity called ‘National Holidays’ with December 25 added to it. The site admin modifies the schedule to add a special day schedule for the special days entity ‘National Holidays’ to override the access and allow janitor to access the facility.

The following example is valid for SpecialDaysSupported=true. The value of Extended-RecurrenceSupported is irrelevant.

```
<SpecialDayGroup token=”8765”>
       <Name>National Holidays</Name>
       <Days>
              BEGIN:VCALENDAR
              PRODID:VERSION:2.0
              BEGIN:VEVENT
              SUMMARY:Christmas day
              DTSTART:20141225T000000
              DTEND:20141226T000000
              UID:Holiday@ONVIF.com
              END:VEVENT
              END:VCALENDAR
       </Days>
<SpecialDayGroup>

<Schedule token=”402”>
       <Name>Access on Monday and Wednesday from 8 AM to 5 PM. No access on Christmas
day</Name>
       <Standard>
              BEGIN:VCALENDAR
              PRODID:VERSION:2.0
              BEGIN:VEVENT
              SUMMARY:Access on Monday and Wednesday from 8 AM to 5 PM
              DTSTART:19700105T080000
              DTEND:19700105T170000
              RRULE:FREQ=WEEKLY;BYDAY=MO,WE
              UID:Event@ONVIF.com
              END:VEVENT
              END:VCALENDAR
       </Standard>
       <SpecialDaysSchedule>
              <GroupToken>8765</GroupToken>
              <TimeRange>
                     <From>00:00:00</From>
              </TimeRange>
       </SpecialDaysSchedule>
</Schedule>
```

A.8 Special Days example 3

A standard schedule gives access on weekdays 7 AM to 6 PM, but not on weekends. A special day starts on Friday, June 10, at 2 PM and ends on Saturday, June 11, at 10 AM. The special days' schedule defines a time range of 8 AM to 4 PM.

```
<SpecialDayGroup token=”7312”>
       <Name>Team building days</Name>
       <Days>
              BEGIN:VCALENDAR
              PRODID:VERSION:2.0
              BEGIN:VEVENT
              SUMMARY:Boot camp
              DTSTART:20160610T140000
              DTEND:20160611T100000
              UID:TeamBuilding@ONVIF.com
              END:VEVENT
              END:VCALENDAR
       </Days>
<SpecialDayGroup>

<Schedule token=”245”>
       <Name>Employee schedule 2016</Name>
       <Description>Access on weekdays from 7 AM to 6 PM.
             Access from 8 AM to 4 PM on Team building days</Description>
       <Standard>
              BEGIN:VCALENDAR
              PRODID:VERSION:2.0
              BEGIN:VEVENT
              SUMMARY:Access on weekdays from 7 AM to 6 PM
              DTSTART:20160101T070000
              DTEND:20161231T180000
              RRULE:FREQ=WEEKLY;BYDAY=MO,TU,WE,TH,FR
              UID:Workhours@ONVIF.com
              END:VEVENT
              END:VCALENDAR
       </Standard>
       <SpecialDaysSchedule>
              <GroupToken>7312</GroupToken>
              <StartTime>08:00:00</StartTime>
              <Until>16:00:00</Until>
       </SpecialDaysSchedule>
</Schedule>
```

Annex B. Revision History

Rev.	Date	Editor	Changes
1.0	Jun 2015	PACS WG	First release
17.06	Jun-2017	Hiroyuki Sano	Change Request 2008, 2009, 2070, 2075, 2076, 2078, 2083
17.12	Dec-2017	Hiroyuki Sano	Change Request 2143, 2168
18.06	Jun-2018	Patrik Björling Rygert	Added support for client-supplied tokens