ONVIF™
Door Control Service Specification

Version 17.12
December 2017

CONTENTS

Contributors

Version 1

Axis Communications AB	Johan Adolfsson
Axis Communications AB	Marcus Johansson
AxxonSoft	Yuri Timenkov
Bosch	Mohane Caliaperoumal
Hirsch Electronics/ Identive Group	Rob Zivney
Honeywell	Marine Drive
Honeywell	Neelendra Bhandari
PACOM	Eugene Scully
PACOM	Steve Barton
Schneider Electric	Mike Berube
Siemens Building Technologies	Klaus Baumgartner
Siemens Building Technologies	Suresh Raman

1 Scope

This specification defines the web service interface for interaction with physical doors. This includes but is not limited to controlling them and monitoring their state.

Web service usage and common ONVIF functionality are outside of the scope of this document. Please refer to the ONVIF Core Specification for those details.

2 Normative references

ONVIF Core Specification
<http://www.onvif.org/specs/core/ONVIF-Core-Specification.pdf>
ONVIF Access Control Specification
<https://www.onvif.org/specs/srv/access/ONVIF-AccessControl-Service-Spec.pdf>

3 Terms and Definitions

3.1 Definitions

Credential	A physical/tangible object, a piece of knowledge, or a facet of a person's physical being, that enables an individual access to a given physical facility or computer-based information system.
Credential (Number)	A sequence of bytes uniquely identifying a credential at an access point.
Door	A physical door, barrier, turnstile, etc. which can be controlled remotely and restricts access between two areas. A door is usually equipped with an electronic lock and a sensor.
Door Alarm	An abnormal state of the door where door is forced open or held open beyond the permitted time duration
Door Lock	A device that secures a door to prevent access, except when explicitly allowed by the access control system. Lock types include electromagnet, electric strike, etc.
Door Mode	Logical state of the door indicating whether the door is locked, unlocked, blocked, locked down or locked open etc.
Door Monitor	Also known as a Door Contact Sensor
Lock	An operation after which a door is locked and alarm is unmasked.
Momentary Access	An operation which invokes the same logic as upon normal access being granted to a credential.
Tamper Detector	Mechanism commonly available for doors, access points and controllers to detect physical tamper
Unlock	An operation to allow a door to be freely used for passage without any door alarms being triggered.

3.2 Abbreviations

ACMS	Access Control Management System
DCU	Door Control Unit
HTTP	Hypertext Transfer (or Transport) Protocol
PACS	Physical Access Control System
REX	Request to exit
TLS	Transport Level Security

4 Overview

The door control service provides mechanisms for controlling physical door instances and monitoring their status.

The Door in this specification can refer to such physical objects as an automatic barrier or a door equipped with electric lock. Turnstiles which can restrict access in either direction can be represented with a pair of doors.

The Door is a subclass of a more generic term Entity defined in the ONVIF Access Control Specification.

Please refer to the ONVIF Access Control specification for generic operation guidelines and design principles behind ONVIF PACS services family.

The service includes the following operations:

- Getting list of doors including their capabilities (e.g., supported operations).
- Getting actual state (e.g., open or closed, locked or unlocked, health status).
- Locking and unlocking.
- Blocking door in locked state such that it can't be accessed.
- Holding door in either unlocked (locked open) or locked (locked down) state and releasing the hold.
- Momentary access.
- Double lock (also known as secure lock) for preventing night-time access.

The service also defines a number of events for real-time monitoring:

- Door physical status change (e.g., open or closed).
- Lock physical state change (e.g., locked or unlocked).
- Operation mode change (e.g., blocked, locked down or locked open).
- Alarm (if door was forced open or was open for too long during momentary access).
- Tamper (an attempt to physically damage its components).
- Hardware malfunction.

5 Door Control

This service offers commands to retrieve status information and to control Door instances of a device.

Please refer to the ONVIF Access Control specification for generic operation guidelines and design principles behind ONVIF PACS services family.

5.1 Service capabilities

An ONVIF compliant device shall provide service capabilities in two ways:

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

2. With the GetServiceCapabilities method.

5.1.1 Data Structures

5.1.1.1 ServiceCapabilities

The ServiceCapabilities structure reflects 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.

5.1.2 GetServiceCapabilities command

This operation returns the capabilities of the service. A device shall support this command.

Table 1: GetServiceCapabilities command

GetServiceCapabilities	Access Class: PRE_AUTH
Message name	**Description**
GetServiceCapabilities Request	*This message shall be empty.*
GetServiceCapabilities Response	*This message contains:* • *"Capabilities": The capability response message contains the requested DoorControl service capabilities using a hierarchical XML capability structure.* tdc:ServiceCapabilities **Capabilities [1][1]** *(extendable)*

5.2 Door information

5.2.1 Data Structures

5.2.1.1 DoorInfo

The DoorInfo type represents the Door as a physical object. The structure contains information and capabilities of a specific door instance. An ONVIF compliant device shall provide the following fields for each Door instance:

- **token**
  A service-unique identifier of the Door.

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

- **Capabilities**
  The capabilities of the Door; is of type DoorCapabilities.

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

- **Description**
  A user readable description. It shall be up to 1024 characters.

5.2.1.2 DoorCapabilities

DoorCapabilities reflect optional functionality of a particular physical entity. Different door instances may have different set of capabilities. This information may change during device operation, e.g. if hardware settings are changed. The following capabilities are available:

- **Access**
  Indicates whether or not this Door instance supports AccessDoor command to perform momentary access.

- **AccessTimingOverride**
  Indicates that this Door instance supports overriding configured timing in the AccessDoor command.

- **Lock**
  Indicates that this Door instance supports LockDoor command to lock the door.

- **Unlock**
  Indicates that this Door instance supports UnlockDoor command to unlock the door.

- **Block**
  Indicates that this Door instance supports BlockDoor command to block the door.

- **DoubleLock**
  Indicates that this Door instance supports DoubleLockDoor command to lock multiple locks on the door.

- **LockDown**
  Indicates that this Door instance supports LockDown (and LockDownRelease) commands to lock the door and put it in LockedDown mode.

- **LockOpen**
  Indicates that this Door instance supports LockOpen (and LockOpenRelease) commands to unlock the door and put it in LockedOpen mode.

- **DoorMonitor**
  Indicates that this Door instance has a DoorMonitor and supports the DoorPhysicalState event.

- **LockMonitor**
  Indicates that this Door instance has a LockMonitor and supports the LockPhysicalState event.

- **DoubleLockMonitor**
  Indicates that this Door instance has a DoubleLockMonitor and supports the DoubleLockPhysicalState event.

- **Alarm**
  Indicates that this Door instance supports door alarm and the DoorAlarm event.

- **Tamper**
  Indicates that this Door instance has a Tamper detector and supports the DoorTamper event.

- **Fault**
  Indicates that this Door instance supports door fault and the DoorFault event.

5.2.2 GetDoorInfoList command

This operation requests a list of all DoorInfo items provided by the device. A device shall support this command.

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 section 4.8.3 of Access Control Service Specification for more details.

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

Table 2: GetDoorInfoList command

GetDoorInfoList	Access Class: READ_SYSTEM
Message name	**Description**
GetDoorInfoListRequest	*This message contains:* • *"Limit": Maximum number of entries to return. If Limit is omitted or if the value of Limit is higher than what the device supports, then the device shall return its maximum amount of entries.* • *"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]**
GetDoorInfoListResponse	*This message contains:* • *"NextStartReference": StartReference to use in next call to get the following items. If absent, no more items to get.* • *"DoorInfo": List of DoorInfo items.* xs:string **NextStartReference [0][1]** tdc:DoorInfo **DoorInfo [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*

5.2.3 GetDoorInfo command

This operation requests a list of DoorInfo items matching the given tokens. A device shall support this command.

The device shall ignore tokens it cannot resolve and may return an empty list if there are no Doors matching specified tokens.

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

Table 3: GetDoorInfo command

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

5.3 Door status

The state of the door may be affected by a number of operations that can be performed on it depending on its capabilities: LockDoor, UnlockDoor, AccessDoor, BlockDoor, DoubleLockDoor, LockDownDoor, LockDownReleaseDoor, LockOpenDoor and LockOpenReleaseDoor.

5.3.1 Data Structures

5.3.1.1 DoorState

The DoorState structure contains current aggregate runtime status of Door.

The following fields are available:

- **DoorPhysicalState**
  Physical state of the Door; it is of type DoorPhysicalState. A device that signals support for DoorMonitor capability for a particular door instance shall provide this field.

- **LockPhysicalState**
  Physical state of the Lock; it is of type LockPhysicalState. A device that signals support for LockMonitor capability for a particular door instance shall provide this field.

- **DoubleLockPhysicalState**
  Physical state of the DoubleLock; it is of type LockPhysicalState. A device that signals support for DoubleLockMonitor capability for a particular door instance shall provide this field.

- **Alarm**
  Alarm state of the door; it is of type DoorAlarmState. A device that signals support for Alarm capability for a particular door instance shall provide this field.

- **Tamper**
  Tampering state of the door; it is of type DoorTamper. A device that signals support for Tamper capability for a particular door instance shall provide this field.

- **Fault**
  Fault information for door; it is of type DoorFault. A device that signals support for Fault capability for a particular door instance shall provide this field.

- **DoorMode**
  The logical operating mode of the door; it is of type DoorMode. An ONVIF compatible device shall report current operating mode in this field.

The following data types define states of DoorState elements.

5.3.1.2 Enumeration: DoorPhysicalState

The physical state of a Door. The following values are available:

- **Unknown**
  Value is currently unknown (possibly due to initialization or monitors not giving a conclusive result).

- **Open**
  Door is open.

- **Closed**
  Door is closed.

- **Fault**
  Door monitor fault is detected.

5.3.1.3 Enumeration: LockPhysicalState

The physical state of a Lock (including Double Lock). The following values are available:

- **Unknown**
  Value is currently not known.

- **Locked**
  Lock is activated.

- **Unlocked**
  Lock is not activated.

- **Fault**
  Lock fault is detected.

5.3.1.4 Enumeration: DoorAlarmState

Describes the state of a Door with regard to alarms. The following values are available:

- **Normal**
  No alarm.

- **DoorForcedOpen**
  Door is forced open.

- **DoorOpenTooLong**
  Door is held open too long.

5.3.1.5 DoorTamper

Tampering information for a Door. The following fields are available:

- **Reason**
  Optional field; Details describing tampering state change (e.g., reason, place and time).

NOTE: All fields (including this one) which are designed to give end-user prompts can be localized to the customer's native language.

- **State**
  State of the tamper detector; it is of type DoorTamperState.

5.3.1.6 Enumeration: DoorTamperState

Describes the state of a Tamper detector. The following values are available:

- **Unknown**
  Value is currently not known.
- **NotInTamper**
  No tampering is detected.
- **TamperDetected**
  Tampering is detected.

5.3.1.7 DoorFault

Fault information for a Door. This can be extended with optional attributes in the future. The following fields are available:

- **Reason**
  Optional reason for fault.
- **State**
  Overall fault state for the door; it is of type DoorFaultState. If there are any faults, the value shall be: FaultDetected. Details of the detected fault shall be found in the Reason field, and/or the various DoorState fields and/or in extensions to this structure.

It can be extended with optional attributes in the future.

5.3.1.8 Enumeration: DoorFaultState

Describes the state of a Door fault. The following values are available:

- **Unknown**
  Fault state is unknown.
- **NotInFault**
  No fault is detected.
- **FaultDetected**
  Fault is detected.

5.3.1.9 Enumeration: DoorMode

DoorMode parameters describe current Door mode from a logical perspective.

The following values are available:

- **Unknown**
  The Door is in an Unknown state.

- **Locked**
  The Door is in a Locked state. In this mode the device shall provide momentary access using the AccessDoor method if supported by the Door instance.

- **Unlocked**
  The Door is in an Unlocked (Permanent Access) state. Alarms related to door timing operations such as open too long or forced are masked in this mode.

- **Accessed**
  The Door is in an Accessed state (momentary/temporary access). Alarms related to timing operations such as “door forced” are masked in this mode.

- **Blocked**
  The Door is in a Blocked state (Door is locked, and AccessDoor requests are ignored, i.e., it is not possible for door to go to Accessed state).

- **LockedDown**
  The Door is in a LockedDown state (Door is locked) until released using the LockDownReleaseDoor command. AccessDoor, LockDoor, UnlockDoor, BlockDoor, and LockOpenDoor requests are ignored, i.e., it is not possible for door to go to Accessed, Locked, Unlocked, Blocked or LockedOpen state.

- **LockedOpen**
  The Door is in a LockedOpen state (Door is unlocked) until released using the LockOpenReleaseDoor command. AccessDoor, LockDoor, UnlockDoor, BlockDoor, and LockDownDoor requests are ignored, i.e., it is not possible for door to go to Accessed, Locked, Unlocked, Blocked or LockedDown state.

- **DoubleLocked**
  The Door is in a Double Locked state - for doors with multiple locks. If the door does not have any DoubleLock, this shall be treated as a normal Locked mode.When changing to an Unlocked mode from the DoubleLocked mode, the door may first go to Locked state before unlocking.

5.3.2 GetDoorState command

This operation requests the state of a Door specified by the Token. A device shall be capable of reporting the status of a door using a DoorState structure available from the GetDoorState command.

Table 4 GetDoorState command

GetDoorState	Access Class: READ_SYSTEM_SENSITIVE
Message name	**Description**
GetDoorStateRequest	*This message contains:* • *"Token": Token of the Door instance to get the state for.* pt:ReferenceToken **Token [1][1]** *(extendable)*
GetDoorStateResponse	*This message contains:* • *"DoorState": The state of the door.* tdc:DoorState **DoorState [1][1]** *(extendable)*
Fault codes	**Description**
env:Sender ter:InvalidArgVal	*The specified token is not found.*

ter:NotFound	

5.4 Door control commands

The service control commands contain operations that allow modifying Door instances states and controlling Door instances of a device.

5.4.1 AccessDoor command

This operation allows momentarily accessing a Door. It invokes the functionality typically used when a card holder presents a card to a card reader at the door and is granted access.

The DoorMode shall change to Accessed state. Please refer to Accessed mode in section 5.3.1.9 for more details.

The Door shall remain accessible for the defined time. When the time span elapses, the DoorMode shall change back to its previous state.

If the request cannot be fulfilled, a Failure fault shall be returned.

Please refer to section 5.3.1.9 for details about Door Modes restrictions.

A device that signals support for Access capability for a particular Door instance shall support this command. A device that signals support for AccessTimingOverride capability for a particular Door instance shall also provide optional timing parameters (AccessTime, OpenTooLongTime and PreAlarmTime) when performing AccessDoor command.

The device shall take the best effort approach for parameters not supported, it must fallback to preconfigured time or limit the time to the closest supported time if the specified time is out of range.

Table 5 AccessDoor command

AccessDoor	Access Class: ACTUATE
Message name	**Description**
AccessDoorRequest	*This message contains:* • *"Token": Token of the Door instance to control.* • *"UseExtendedTime": Optional - Indicates that the configured extended time should be used.* • *"AccessTime": Optional - overrides AccessTime if specified.* • *"OpenTooLongTime": Optional - overrides OpenTooLongTime if specified (DOTL).* • *"PreAlarmTime": Optional - overrides PreAlarmTime if specified.* • *"Extension": Future extension.* pt:ReferenceToken **Token [1][1]** xs:boolean **UseExtendedTime [0][1]** xs:duration **AccessTime [0][1]** xs:duration **OpenTooLongTime [0][1]** xs:duration **PreAlarmTime [0][1]** tdc:AccessDoorExtension **Extension [0][1]**
AccessDoorResponse	*This message is typically empty, but is extendable*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*
env:Receiver	*Failed to go to Accessed state and unlock the door.*

ter:Action ter:Failure	

5.4.2 LockDoor command

This operation allows locking a Door. The DoorMode shall change to Locked state. Please refer to Locked mode in section 5.3.1.9 for more details.

A device that signals support for Lock capability for a particular Door instance shall support this command.

If the request cannot be fulfilled, a Failure fault shall be returned. Please refer to section 5.3.1.9 for more details about Door Modes restrictions.

Table 6 LockDoor command

LockDoor	Access Class: ACTUATE
Message name	**Description**
LockDoorRequest	*This message contains:* • *"Token": Token of the Door instance to control.* pt:ReferenceToken **Token [1][1]** *(extendable)*
LockDoorResponse	*This message is typically empty, but is extendable*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*
env:Receiver ter:Action ter:Failure	*Failed to go to Locked state.*

5.4.3 UnlockDoor command

This operation allows unlocking a Door. The DoorMode shall change to Unlocked state.Please refer to Unlocked mode in section 5.3.1.9 for more details.

A device that signals support for Unlock capability for a particular Door instance shall support this command.

If the request cannot be fulfilled, a Failure fault shall be returned.Please refer to section 5.3.1.9 for more details about Door Modes restrictions.

Table 7 UnlockDoor command

UnlockDoor	Access Class: ACTUATE
Message name	**Description**
UnlockDoorRequest	*This message contains:* • *"Token": Token of the Door instance to control.* pt:ReferenceToken **Token [1][1]** *(extendable)*
UnlockDoorResponse	*This message is typically empty, but is extendable*

Fault codes	Description
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*
env:Receiver ter:Action ter:Failure	*Failed to go to Unlocked state.*

5.4.4 BlockDoor command

This operation allows blocking a Door and preventing momentary access (AccessDoor command). The DoorMode shall change to Blocked state.Please refer to Blocked mode in section 5.3.1.9 for more details.

A device that signals support for Block capability for a particular Door instance shall support this command.

If the request cannot be fulfilled, a Failure fault shall be returned. Please refer to section 5.3.1.9 for more details about Door Modes restrictions.

Table 8 BlockDoor command

BlockDoor	Access Class: ACTUATE
Message name	**Description**
BlockDoorRequest	*This message contains:* • *"Token": Token of the Door instance to control.* pt:ReferenceToken **Token [1][1]** *(extendable)*
BlockDoorResponse	*This message is typically empty, but is extendable*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*
env:Receiver ter:Action ter:Failure	*Failed to go to Blocked state.*

5.4.5 LockDownDoor command

This operation allows locking and preventing other actions until a LockDownRelease command is invoked. The DoorMode shall change to LockedDown state. Please refer to LockedDown mode in section 5.3.1.9 for more details.

The device shall ignore other door control commands until a LockDownRelease command is performed.

A device that signals support for LockDown capability for a particular Door instance shall support this command.

If a device supports DoubleLock capability for a particular Door instance, that operation may be engaged as well.

If the request cannot be fulfilled, a Failure fault shall be returned. Please refer to section 5.3.1.9 for more details about Door Modes restrictions.

Table 9 LockDownDoor command

LockDownDoor	Access Class: ACTUATE
Message name	**Description**
LockDownDoorRequest	*This message contains:* • *"Token": Token of the Door instance to control.* pt:ReferenceToken **Token [1][1]** *(extendable)*
LockDownDoorResponse	*This message is typically empty, but is extendable*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*
env:Receiver ter:Action ter:Failure	*Failed to go to a LockedDown state.*

5.4.6 LockDownReleaseDoor command

This operation allows releasing the LockedDown state of a Door. The DoorMode shall change back to its previous/next state. It is not defined what the previous/next state shall be, but typically - Locked. A device that signals support for LockDown capability for a particular Door instance shall support this command.

This method shall only succeed if the current DoorMode is LockedDown.

Table 10 LockDownReleaseDoor command

LockDownReleaseDoor	Access Class: ACTUATE
Message name	**Description**
LockDownReleaseDoorRequest	*This message contains:* • *"Token": Token of the Door instance to control.* pt:ReferenceToken **Token [1][1]** *(extendable)*
LockDownReleaseDoorResponse	*This message is typically empty, but is extendable*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*
env:Receiver ter:Action ter:Failure	*Failed to leave LockedDown state.*

5.4.7 LockOpenDoor command

This operation allows unlocking a Door and preventing other actions until LockOpenRelease method is invoked. The DoorMode shall change to LockedOpen state. Please refer to LockedOpen mode in section 5.3.1.9 for more details.

The device shall ignore other door control commands until a LockOpenRelease command is performed.

A device that signals support for LockOpen capability for a particular Door instance shall support this command.

If the request cannot be fulfilled, a Failure fault shall be returned. Please refer to section 5.3.1.9 for more details about Door Modes restrictions.

Table 11 LockOpenDoor command

LockOpenDoor	Access Class: ACTUATE
Message name	**Description**
LockOpenDoorRequest	*This message contains:* • *"Token": Token of the Door instance to control.* pt:ReferenceToken **Token [1][1]** *(extendable)*
LockOpenDoorResponse	*This message is typically empty, but is extendable*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*
env:Receiver ter:Action ter:Failure	*Failed to go to LockedOpen state.*

5.4.8 LockOpenReleaseDoor command

This operation allows releasing the LockedOpen state of a Door. The DoorMode shall change state from the LockedOpen state back to its previous/next state. It is not defined what the previous/next state shall be, but typically - Unlocked. A device that signals support for LockOpen capability for a particular Door instance shall support this command.

This method shall only succeed if the current DoorMode is LockedOpen.

Table 12 LockOpenReleaseDoor command

LockOpenReleaseDoor	Access Class: ACTUATE
Message name	**Description**
LockOpenReleaseDoorRequest	*This message contains:* • *"Token": Token of the Door instance to control.* pt:ReferenceToken **Token [1][1]** *(extendable)*
LockOpenReleaseDoorResponse	*This message is typically empty, but is extendable*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*
env:Receiver	*Failed to leave LockedOpen state.*

ter:Action ter:Failure	

5.4.9 DoubleLockDoor command

This operation is used for securely locking a Door. A call to this method shall change DoorMode state to DoubleLocked. Please refer to DoubleLocked mode in section 5.3.1.9 for more details.

A device that signals support for DoubleLock capability for a particular Door instance shall support this command. Otherwise this method can be performed as a standard Lock operation (see 5.4.2 LockDoor command).

If the door has an extra lock that shall be locked as well.

If the request cannot be fulfilled, a Failure fault shall be returned.

Table 13 DoubleLockDoor command

DoubleLockDoor	Access Class: ACTUATE
Message name	**Description**
DoubleLockDoorRequest	*This message contains:* • *"Token": Token of the Door instance to control.* pt:ReferenceToken **Token [1][1]** *(extendable)*
DoubleLockDoorResponse	*This message is typically empty, but is extendable*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*
env:Receiver ter:Action ter:Failure	*Failed to go to DoubleLocked state.*

6 Notification Topics

This section defines notification topics specific to Door Control service.

Please refer to Access Control specification for generic operation guidelines and design principles behind ONVIF PACS services family.

6.1 Status changes

Whenever a door mode is changed, the device shall provide the following event:

```
Topic: tns1:Door/State/DoorMode

<tt:MessageDescription IsProperty="true">
<tt:Source>
<tt:SimpleItemDescription Name="DoorToken" Type="pt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:SimpleItemDescription Name="State" Type="tdc:DoorMode"/>
</tt:Data>
</tt:MessageDescription>
```

A device that signals support for DoorMonitor capability for a particular door instance shall provide the following event whenever the physical state of this door is changed:

```
Topic: tns1:Door/State/DoorPhysicalState

<tt:MessageDescription IsProperty="true">
<tt:Source>
<tt:SimpleItemDescription Name="DoorToken" Type="pt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:SimpleItemDescription Name="State" Type="tdc:DoorPhysicalState"/>
</tt:Data>
</tt:MessageDescription>
```

A device that signals support for LockMonitor capability for a particular door instance shall provide the following event whenever the physical state of this door's lock is changed:

```
Topic: tns1:Door/State/LockPhysicalState

<tt:MessageDescription IsProperty="true">
<tt:Source>
<tt:SimpleItemDescription Name="DoorToken" Type="pt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:SimpleItemDescription Name="State" Type="tdc:LockPhysicalState"/>
</tt:Data>
</tt:MessageDescription>
```

A device that signals support for DoubleLockMonitor capability for a particular door instance shall provide the following event whenever the physical state of this door's secure lock is changed:

```
Topic: tns1:Door/State/DoubleLockPhysicalState

<tt:MessageDescription IsProperty="true">
<tt:Source>
<tt:SimpleItemDescription Name="DoorToken" Type="pt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:SimpleItemDescription Name="State" Type="tdc:LockPhysicalState"/>
</tt:Data>
</tt:MessageDescription>
```

A device that signals support for Alarm capability for a particular door instance shall provide the following event whenever the alarm state of this door is changed:

```
Topic: tns1:Door/State/DoorAlarm

<tt:MessageDescription IsProperty="true">
<tt:Source>
<tt:SimpleItemDescription Name="DoorToken" Type="pt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:SimpleItemDescription Name="State" Type="tdc:DoorAlarmState"/>
</tt:Data>
</tt:MessageDescription>
```

A device that signals support for Tamper capability for a particular door instance shall provide the following event whenever the tamper state of this door is changed:

```
Topic: tns1:Door/State/DoorTamper

<tt:MessageDescription IsProperty="true">
<tt:Source>
<tt:SimpleItemDescription Name="DoorToken" Type="pt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:SimpleItemDescription Name="State" Type="tdc:DoorTamperState"/>
</tt:Data>
</tt:MessageDescription>
```

A device that signals support for Fault capability for a particular door instance shall provide the following event whenever the fault state of this door is changed:

```
Topic: tns1:Door/State/DoorFault

<tt:MessageDescription IsProperty="true">
<tt:Source>
<tt:SimpleItemDescription Name="DoorToken" Type="pt:ReferenceToken"/>
</tt:Source>
<tt:Data>
<tt:SimpleItemDescription Name="State" Type="tdc:DoorFaultState"/>
<tt:SimpleItemDescription Name="Reason" Type="xs:string"/>
</tt:Data>
</tt:MessageDescription>
```

The Reason element may be empty or absent. The device may also skip it unless the fault state is FaultDetected.

6.2 Configuration changes

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

```
Topic: tns1:Configuration/Door/Changed

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

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

```
Topic: tns1:Configuration/Door/Removed

<tt:MessageDescription IsProperty="false">
<tt:Source>
<tt:SimpleItemDescription Name="DoorToken" Type="pt:ReferenceToken"/>
</tt:Source>

</tt:MessageDescription>
```

Annex A. Revision History

Rev.	Date	Editor	Changes
1.0	Apr-2013	Yuri Timenkov	First Version
1.0.1	Aug-2013	Hans Busch	Change Request 1053
1.0.2	Jun-2014	Michio Hirai	Change Request 1368, 1369
17.12	Dec-2017	Hiroyuki Sano	Change Request 2172, 2173