

ONVIF™
Application Management
Service Specification

Version 19.12

December, 2019



© 2008-2019 by ONVIF: Open Network Video Interface Forum Inc.. All rights reserved.

Recipients of this document may copy, distribute, publish, or display this document so long as this copyright notice, license and disclaimer are retained with all copies of the document. No license is granted to modify this document.

THIS DOCUMENT IS PROVIDED "AS IS," AND THE CORPORATION AND ITS MEMBERS AND THEIR AFFILIATES, MAKE NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, OR TITLE; THAT THE CONTENTS OF THIS DOCUMENT ARE SUITABLE FOR ANY PURPOSE; OR THAT THE IMPLEMENTATION OF SUCH CONTENTS WILL NOT INFRINGE ANY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.

IN NO EVENT WILL THE CORPORATION OR ITS MEMBERS OR THEIR AFFILIATES BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE OR CONSEQUENTIAL DAMAGES, ARISING OUT OF OR RELATING TO ANY USE OR DISTRIBUTION OF THIS DOCUMENT, WHETHER OR NOT (1) THE CORPORATION, MEMBERS OR THEIR AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, OR (2) SUCH DAMAGES WERE REASONABLY FORESEEABLE, AND ARISING OUT OF OR RELATING TO ANY USE OR DISTRIBUTION OF THIS DOCUMENT. THE FOREGOING DISCLAIMER AND LIMITATION ON LIABILITY DO NOT APPLY TO, INVALIDATE, OR LIMIT REPRESENTATIONS AND WARRANTIES MADE BY THE MEMBERS AND THEIR RESPECTIVE AFFILIATES TO THE CORPORATION AND OTHER MEMBERS IN CERTAIN WRITTEN POLICIES OF THE CORPORATION.

CONTENTS

- 1 Scope** **4**
- 2 Normative references** **4**
- 3 Informative references** **4**
- 4 Terms and Definitions** **4**
 - 4.1 Definitions.....4
 - 4.2 Abbreviations4
- 5 Overview** **5**
- 6 Service** **6**
 - 6.1 General.....6
 - 6.2 Application Deployment6
 - 6.2.1 Install.....6
 - 6.2.2 Update6
 - 6.2.3 Uninstall6
 - 6.2.4 Activate7
 - 6.2.5 Deactivate7
 - 6.3 Application Information.....7
 - 6.3.1 Information Items7
 - 6.3.2 GetInstalledApps8
 - 6.3.3 GetAppsInfo8
 - 6.4 Licensing9
 - 6.4.1 InstallLicense9
 - 6.5 GetCapabilities9
- 7 Events** **10**
 - 7.1 State Change10
- Annex A. Revision History** **11**

1 Scope

This document defines an interface for managing applications on a device. This specification focuses on the network interface for managing the application lifecycle on a device. It includes installation, information retrieval as well as controlling execution. The actual runtime environment as well as the application packaging format are out of scope of this specification.

2 Normative references

ONVIF Core Specification

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

3 Informative references

Docker Enterprise Application Container Platform

< <https://www.docker.com/> >

Android

< <https://www.android.com/> >

4 Terms and Definitions

4.1 Definitions

App Application that can be installed and execute on a networked device.

4.2 Abbreviations

HTTP Hypertext Transfer Protocol
TLS Transport Layer Security

5 Overview

A variety of standardized and proprietary application runtime formats exist. Examples for well-known standardized runtimes are Docker and Android. This specification provides network interface APIs based on the ONVIF Core Specification for uploading and controlling applications on a networked device.

The following basic functionalities are defined by this specification:

- Installing and uninstalling of applications.
- Starting and stopping execution of applications.
- Enumeration of installed applications
- Listing of application details
- Enrolling application licenses

An application state change event allows client to retrieve and track application state changes without need for polling.

Note that both the container format as well as the runtime are outside of the scope of this specification.

6 Service

6.1 General

Each application has an associated state. Possible states are Installing, Active, Inactive, DeInstalling and InstallationFailed. Clients can retrieve the application state via the GetAppInfo method or by subscribing to the application state property event. The latter mechanism allows retrieving the initial state as well as updates via property events.

6.2 Application Deployment

6.2.1 Install

Applications are deployed to a device via http POST to the URI provided by GetServiceCapabilities. The same URI shall be used for updating an application.

A device supporting this service shall support the POST operation to the upload URI provided by the UploadUri capability.

When posting an image to one of the URIs, the following HTML status codes may be returned, depending on success or failure:

- 200 OK
image file was successfully uploaded
- 401 Unauthorized
attempted POST without authentication credentials at the WRITE_SYSTEM access policy level
- 408 Request Timeout
POST took too long to upload
- 411 Length Required
POST does not include Content-Length header
- 415 Unsupported Media Type
POST does not include Content-Type header, or file MIME type does not match any type in the FormatsSupported attribute's list

Note that the installation may not be completed when the upload has finished.

If uploading succeeded, a device shall signal start of the installation via the state property event. If the installation fails, the device shall send a final InstallationFailed state. Otherwise the state shall change to either Active or Inactive.

6.2.2 Update

For updates, a device shall support the same mechanism as the installation procedure defined in 6.1.1.

In case an application is in state Active, the device shall change its state to Inactive before signaling installation state change events as stated in 6.1.1.

6.2.3 Uninstall

A device supporting this service shall support this method to stop and remove an installed application.

REQUEST:

- **AppID** – [xs:string]
Application to be removed

RESPONSE:

<empty>

FAULTS:

- **env:Sender - ter:InvalidArgVal – ter:UnkownId**
The requested application ID does not exist.

ACCESS CLASS:

SYSTEM_WRITE

6.2.4 Activate

A device supporting this service shall support this method to start an installed application.

REQUEST:

- **AppID – [xs:string]**
Application to be started

RESPONSE:

<empty>

FAULTS:

- **env:Sender - ter:InvalidArgVal – ter:UnkownId**
The requested application ID does not exist.

ACCESS CLASS:

SYSTEM_WRITE

6.2.5 Deactivate

A device supporting this service shall support this method to stop an installed application.

REQUEST:

- **AppID – [xs:string]**
Application to be stopped

RESPONSE:

<empty>

FAULTS:

- **env:Sender - ter:InvalidArgVal – ter:UnkownId**
The requested application ID does not exist.

ACCESS CLASS:

SYSTEM_WRITE

6.3 Application Information

6.3.1 Information Items

The following items may be associated to each application and can be queried either via the GetInstalledApps method or the GetAppInfo method.

- **AppID** Unique identifier of an application
- **Name** User readable name of the application
- **Version** Version string assigned by the application vendor
- **Licenses** Licenses associated with the application
- **Privileges** Privileges assigned to the application

- **InstallationDate** Date when the application was installed
- **LastUpdate** Date when the application was last updated
- **State** Information whether the application is currently Installing, Active, Inactive or Uninstalling (see Figure 1).
- **Status** Textual description of the state. In error cases this shall contain the error reason.
- **Autostart** Indication whether the application automatically starts on system boot.
- **Website** Link to an internal or external website providing supplementary information about the application or its vendor.
- **OpenSource** Optional link to a listing of related open source licenses.

6.3.2 GetInstalledApps

A device supporting this service shall support this command to enumerate all installed applications. This method provides only high level information on the applications for more detailed information see GetAppInfo.

REQUEST:

<empty>

RESPONSE:

- **App – optional, unbounded [Application]**
List of applications containing its ID as well as a user readable name.

FAULTS:

<none>

ACCESS CLASS:

SYSTEM_READ

6.3.3 GetAppsInfo

A device supporting this service shall support this command to retrieve detailed information about applications. The caller may provide an application ID to retrieve the information for a single application. If no application ID is provided, the device shall report the information for all installed applications.

REQUEST

- **AppID – optional [xs:string]**
Optional ID to only retrieve information for a single application

RESPONSE

- **Info – optional, unbounded [AppInfo]**
List of detailed information about the applications.

FAULTS:

- **env:Sender - ter:InvalidArgVal – ter:UnkownId**
The requested application ID does not exist.

ACCESS CLASS:

SYSTEM_READ

6.4 Licensing

6.4.1 InstallLicense

A device signaling support via the Licensing capability shall support this command to install a license on the device. The format and structure of the license string is outside of the scope of this specification.

Refer to GetAppInfo for listing installed application licenses.

REQUEST

- **AppID – optional [xs:string]**
Application the license shall be associated to.
- **License – [xs:string]**
Computer readable license string to be installed.

RESPONSE

<empty>

FAULTS:

- **env:Sender - ter:InvalidArgVal – ter:InvalidFormat**
The device cannot interpret the license either because the format is unknown or because it has invalid syntax.

ACCESS CLASS:

SYSTEM_WRITE

6.5 GetCapabilities

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

UploadUri Mandatory upload URI for applications

Licensing Signals support for licensing of applications

SupportedFormat List of application formats supported by the device.

REQUEST:

This message is empty

RESPONSE:

- **Capabilities [timg:Capabilities]**
Set of indicators for function groups as described above.

FAULTS:

None

ACCESS CLASS:

PRE_AUTH

7 Events

7.1 State Change

Whenever the state of an application instance changes the following property event shall be generated. Figure 1 shows the possible states and their transitions.

tns1:AppMgmt/State

```
<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItemDescription Name="AppID" Type="xs:string"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="State" Type="tam:State"/>
    <tt:SimpleItemDescription Name="Status" Type="xs:string"/>
  </tt:Data>
</tt:MessageDescription>
```

The Status item is optional. It shall be included whenever an error triggered the state change.

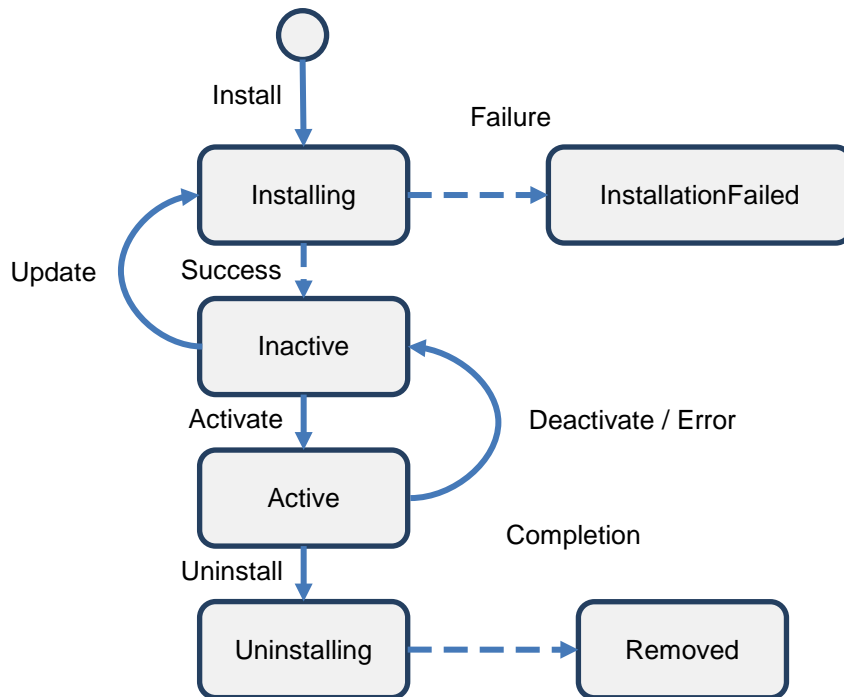


Figure 1: Application state chart.

Annex A. Revision History

Rev.	Date	Editor	Changes
19.12	Dec-2019	Hans Busch	Initial release