ONVIF™
Recording Search Specification

Version 26.06

June, 2026

CONTENTS

1 Scope

This document defines the web service interface for searching for recorded Video, Audio and Metadata.

For a definition of the storage model see the ONVIF Recording Control Specification.

Web service usage is outside of the scope of this document. Please refer to the ONVIF core specification.

2 Normative references

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

ONVIF Recording Control Specification
<http://www.onvif.org/specs/srv/rec/ONVIF-RecordingControl-Service-Spec.pdf>

W3C XML Path Language (XPath) Version 1.0
<https://www.w3.org/TR/1999/REC-xpath-19991116>

3 Terms and Definitions

3.1 Definitions

Metadata	All streaming data except video and audio, including video analytics results, PTZ position data and other metadata (such as textual data from POS applications).
Recording	A container for a set of audio, video and metadata tracks. A recording can hold one or more tracks. A track is viewed as an infinite timeline that holds data at certain times.
Recording Event	An event associated with a Recording, represented by a notification message in the APIs
Recording Job	A job performs the transfer of data from a data source to a particular recording using a particular configuration
Track	An individual data channel consisting of video, audio, or metadata. This definition is consistent with the definition of track in [RFC 2326]
Video Analytics	Algorithms or programs used to analyze video data and to generate data describing object location and behaviour.

4 Overview

The search service enables a client to find information about the recordings on the storage device, for example to construct a “timeline” view, and to find data of interest within a set of recordings. The latter is achieved by searching for events and other information that is included in the metadata track recording.

The search service provides the following functionality:

- Find recordings and information about each recording
- Find events in the metadata and among the historical events
- Find PTZ positions in the metadata
- Find other information in the metadata e.g. text from EPOS (electronic point-of-sale) systems

The actual searching is done by coupled find and result operations and is asynchronous. Each find operation initiates a search session. The client can then acquire the results from the search session in increments, or all

at once, depending on implementation and the scale of the search. There are four pairs of search operations for recordings, recording events, PTZ positions and metadata.

FindRecordings and GetRecordingSearchResults

FindEvents and GetEventSearchResults

FindPTZPosition and GetPTZPositionSearchResults

FindMetadata and GetMetadataSearchResults

WSDL for this service is specified in http://www.onvif.org/ver10/search.wsdl.

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
tse	http://www.onvif.org/ver10/search/wsdl

5 Service

5.1 Introduction

The search service provides a number of operations for finding data of interest within a set of recordings. The most common way of doing this would be to search for events that are either included in the metadata track of a recording, or are otherwise associated with a recording in the device (see Recording Events below).

GetRecordingSummary returns a summary for all recordings, and can be used to provide the scale of a timeline.

GetRecordingInformation returns information about a single recording, such as start time and current status.

GetMediaAttributes returns the media attributes of a recording at a specific point in time.

The actual search is done by coupled find and result operations. Each find operation initiates a search session. The client can then acquire the results from the search session in increments, or all at once, depending on implementation and the scale of the search. There are four pairs of search operations for Recordings, Recording Events, PTZ Positions and Metadata.

GetSearchState returns the state of a search session.

EndSearch ends a search session, halting search and returning any blocking result operations.

5.2 Concepts

5.2.1 Search Direction

Search is performed from a start point on the time line, towards an end point. If the end point is prior to the start point, search will be performed backwards. This can be useful if only the newest matching event is of interest, or if it is otherwise convenient to get the results in newest to oldest order.

If no end point is specified, the search will always be performed forward in time from the start point.

5.2.2 Recording Event

Describes a discrete event related to the recording. It is represented as a notification message, but this does not necessarily mean it has been recorded as a notification. Recording events can either be notifications included

in a recorded metadata track, it can be created by the recording device as a result of an internal event or mechanism, or it can be inserted by a client using a WebService request or a metadata stream. However the recording event has been created and associated with a particular recording, this specification makes no implications on how it is stored internally on a device, only how it should be represented in the interface.

However created, recording events are always treated as notifications in regards to search filters and results returned. Each recording event has a notification topic as defined in the Topic Structure section of the ONVIF Core Specification. Predefined recording events are described in section 5.21.

To communicate the original state of property events, virtual start state events can be returned in a search result containing the value of one or more properties at the start point of the search interval. Such start state events are virtual events in the sense that they are created on the fly by the server, rather than being collected from recorded data. If the client indicates that such events are desired by setting the appropriate flag, virtual events matching the topics defined in the search filter shall be returned for any recording in the search scope.

5.2.3 Search Session

A search session is started asynchronously by a Find-operation and is identified by a search token unique for that session. Results are returned in increments using GetResult-operations referring to the session created by the Find-operation. The search can be terminated in three ways:

- KeepAlive time expires – If no request from a client has been made that refers to a particular session within the specified time interval, it will terminate.
- A GetResult method returns the last data for the search session by setting the search state in its result to “Completed”.
- EndSearch – The client explicitly ends a session.

Ending a session will cancel an ongoing search, immediately return and make further requests to the same session result in an error message. A device shall not reuse a search token immediately as it would confuse clients unaware that a session had ended.

5.2.4 Search Scope

The scope contains a number of optional elements, together limiting the set of data to look into when performing searches.

5.2.4.1 Included data

Optionally, the client can define sources and recordings to search in by specifying lists of tokens for each type. If several types are given, the union of the specified tokens shall be used. If there are no sources or recording tokens specified, all recordings shall be included. The scope is further refined by the Recording Information Filter. However, if recordings are specified the filters will only be applied to that subset of recordings.

5.2.4.2 Recording Information Filter

Rather than specifying a list of recording tokens, the recordings can be filtered by an Xpath filter operating on the RecordingInformation structure. This allows the client to filter on all elements present in the RecordingInformation structure, using comparisons according to the Xpath dialect defined in section 5.22. If a recording information filter is supplied, only recordings matching the filter shall be part of the scope.

Example of a filter that includes only recordings containing audio in the search scope:

```
boolean(//Track[TrackType = “Audio”])
```

5.2.5 Search Filters

Search filters are specific for the type of search operation. See FindEvents, FindPTZPosition, FindMetadata respectively. They all act on the recordings defined by the scope.

5.2.6 Time Information

An ONVIF compliant device shall support time values in request parameters that are given in utc with the 'Z' indicator and respond all time values as utc including the 'Z' indicator.

5.3 Data Structures

5.3.1 RecordingInformation Structure

RecordingInformation contains information about a recording, the tracks it consists of and the source.

RecordingToken a unique identifier of the recording.

EarliestRecording the date and time of the oldest data in the recording

LatestRecording the date and time of the newest data in the recording.

Content informative description of content. If the content is unknown this field is empty

RecordingStatus current status of recording, can be any of: Initiated, Recording, Stopped, Removing, Removed, Unknown.

RecordingSourceInformation a structure containing information about the source of the recording.

TrackInformation a list of track information structures.

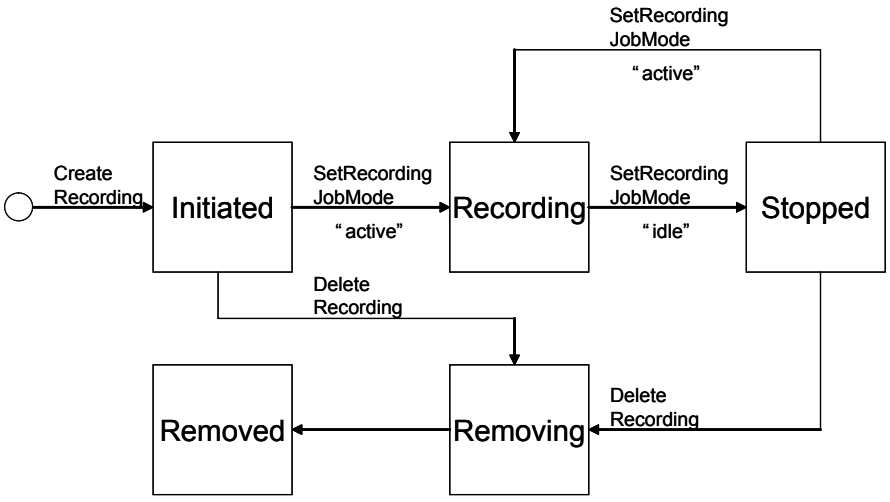

Figure 1: Recording state chart.

Figure 1 shows the state changes for the RecordingStatus. The following adds additional explanation:

Initiated The new recording is created and the specified recording has not yet started to record.

Recording The recording job is executing the specified recording.

Stopped The recording job is pausing the specified recording. This case requires that the job has started recording before at least once.

Removing The specified recording is in the process of being deleted.

Removed The specified recording has been removed.

Unknown This case should never happen.

5.3.2 RecordingSourceInformation Structure

Contains information about the source of a recording.

SourceId an identifier for the source chosen by the client that creates the recording. This identifier is opaque to the device. Clients may use any type of URI for this field. If this identifier is not present this field is empty.

Name informative name of the source. If the name is unknown this field is empty.

Location informative description of the location of the source. If the description is not available this string is empty.

Description informative description of the source. If this description is not present, the description is empty.

Address informative URI of the source.

5.3.3 TrackInformation Structure

Contains information about a single track in a recording. Note that a track may represent a single contiguous time span or consist of multiple slices as shown in the introduction of the ONVIF Recording Control Specification. If there is no recorded data for a track the TrackInformation shall not be provided.

TrackToken An identifier of the track. The TrackToken is unique between all TrackTokens used within a recording.

TrackType Identifies the type of track (video, audio or metadata)

Description Informative description of the track. If this information is not present, this field is empty.

DataFrom The start date and time of the oldest recorded data in the track.

DataTo The stop date and time of the newest recorded data in the track.

5.3.4 SearchState Enumeration

The search state can be one of the following

Searching The database search is in progress and there may be results available that can be fetched via the method GetEventSearchResults.

Completed Search has been completed and all results have been delivered via GetEventSearchResults.

The usage of the additionally defined state "Queued" is deprecated.

5.3.5 MediaAttributes Structure

The MediaAttributes contains information about the media tracks of a particular recording for a particular time frame. The time frame can be a single point in time, in which case the *From* and *Until* elements are identical.

RecordingToken A reference to the recording that this structure concerns.

From A point in time from when the attributes are valid for the recording.

Until A point in time until when the specified attributes are valid for the recording.

VideoAttributes A set of video attributes, describing the data of a recorded video track.

AudioAttributes A set of audio attributes, describing the data of a recorded audio track.

MetadataAttributes A set of attributes, describing the possible metadata content of a recorded metadata track.

5.3.6 FindEventResult Structure

RecordingToken Identifying the recording containing the found event.

TrackToken Identifying the track containing the found event.

Time The date and time of the found event.

Event The event message found.

StartStateEvent If true, indicates the event represents the start state of one or more properties in the recording.

5.3.7 FindPTZPositionResult Structure

RecordingToken – identifying the recording containing the matching position.

TrackToken Identifying the track containing the matching position.

Time The date and time of the matching position.

Position The matching PTZ vector.

5.3.8 PTZPositionFilter Structure

Contains the necessary elements to define what PTZ positions to search for. The PTZ vectors shall be in the same coordinate space as the PTZ coordinates stored in the recording.

MinPosition The lower boundary of the PTZ volume to look for.

MaxPosition The upper boundary of the PTZ volume to look for.

EnterOrExit If true, report the positions when entering or exiting the specified PTZ volume. Otherwise report all recorded positions within the specified PTZ range.

5.3.9 MetadataFilter Structure

Contains an Xpath expression to be applied to the MetadataStream structure.

Example of an expression searching for objects overlapping the lower right quadrant of the scene:

```
boolean(//Object/Appearance/Shape/BoundingBox[@right > ”0.5”]) and
boolean(//Object/Appearance/Shape/BoundingBox[@bottom > ”0.5”])
```

5.3.10 FindMetadataResult Structure

RecordingToken Identifying the recording containing the matching metadata.

TrackToken Identifying the track containing the matching metadata.

Time The date and time of the matching metadata.

5.4 GetRecordingSummary

GetRecordingSummary is used to get a summary description of all recorded data. This operation is mandatory to support for a device implementing the recording search service. If a device returns NumberRecordings as zero, both DataFrom and DataUntil can be safely ignored.

REQUEST:

This is an empty message.

RESPONSE:

- **Summary [tt:RecordingSummary]**
  A structure containing: DataFrom specifying the first time when there is recorded data on the device; DataUntil specifying the last point in time where there is data recorded on the device; the total number of recordings on the device.

FAULTS:

None

ACCESS CLASS:

READ_MEDIA

5.5 GetRecordingInformation

Returns information about a single *Recording* specified by a *RecordingToken.* This operation is mandatory to support for a device implementing the recording search service.

REQUEST:

- **RecordingToken [tt:ReferenceToken]**
  Identifies the recording.

RESPONSE:

- **RecordingInformation [tt:RecordingInformation]**
  Information about the recording.

FAULTS:

- **env:Sender - ter:InvalidArgVal - ter:InvalidToken**
  The recording token is not valid

ACCESS CLASS:

READ_MEDIA

5.6 GetMediaAttributes

Returns a set of media attributes for all tracks of the specified recordings at a specified point in time. Clients using this operation shall be able to use it as a non-blocking operation. A device shall set the starttime and endtime of the MediaAttributes structure to equal values if calculating this range would cause this operation to block. See MediaAttributes structure for more information. This operation is mandatory to support for a device implementing the recording search service.

Devices indicating CanContainPTZ shall report the PTZ spaces in use at the specified point in time (including generic spaces). For optimal interoperability device implementations should use generic spaces. The generic spaces are the following:

```
http://www.onvif.org/ver10/tptz/PanTiltSpaces/PositionGenericSpace
http://www.onvif.org/ver10/tptz/ZoomSpaces/PositionGenericSpace
```

REQUEST:

- **RecordingTokens – optional, unbounded [tt:ReferenceToken]**
  A list of references to the recordings to query. If no recording tokens are provided all recordings should be queried.
- **Time [xs:dateTime]**
  The point in time from where the information is requested.

RESPONSE:

- **MediaAttributes – optional, unbounded [tt:MediaAttributes]**
  Contains a MediaAttributes structure for the RecordingToken specified in the request. Note that each RecordingToken can result in zero or one MediaAttributes.

FAULTS:

- **env:Sender - ter:InvalidArgVal - ter:InvalidToken**
  The recording token is not valid.

ACCESS CLASS:

READ_MEDIA

5.7 FindRecordings

FindRecordings starts a search session, looking for recordings that match the scope (See 5.2.4) defined in the request. Results from the search are acquired using the GetRecordingSearchResults request, specifying the search token returned from this request.

The device shall continue searching until one of the following occurs:

- The total number of matches has been found, defined by the *MaxMatches* parameter.
- The session has been ended by a client EndSearch request.
- The session has been ended because *KeepAliveTime* since the last request related to this session has expired.

The order of the results is undefined, to allow the device to return results in any order they are found. This operation is mandatory to support for a device implementing the recording search service.

For the KeepAliveTime a device shall support at least values up to ten seconds. A device may adapt larger values.

REQUEST:

- **Scope [tt:SearchScope]**
  Defines the dataset to consider for this search
- **MaxMatches - optional [xs:int]**
  The search ends after MaxMatches.
- **KeepAliveTimeout [xs:duration]**
  The session timeout after each request concerning this session.

RESPONSE:

- **SearchToken [tt:Jobtoken]**
  Identifies the search session created by this request.

FAULTS:

- **env:Sender - ter:InvalidArgVal - ter:InvalidToken**
  The recording token is not valid.
- **env:Sender - ter:InvalidArgVal - ter:InvalidSource**
  The recording source is not valid.
- **env:Receiver - ter:Action - ter:ResourceProblem**
  Device is unable to create a new search session.

ACCESS CLASS:

READ_MEDIA

5.8 GetRecordingSearchResults

GetRecordingSearchResults acquires the results from a recording search session previously initiated by a FindRecordings operation. The response shall not include results already returned in previous requests for the same session. If *MaxResults* is specified, the response shall not contain more than *MaxResults* results. The number of results relates to the number of recordings. For viewing individual recorded data for a single track use the FindEvents method.

GetRecordingSearchResults shall block until:

- *MaxResults* results are available for the response if *MaxResults* is specified.
- *MinResults* results are available for the response if *MinResults* is specified.
- *WaitTime* has expired.
- Search is completed or stopped.

This operation is mandatory to support for a device implementing the recording search service. If any of the specified parameters MinResults and WaitTime exceed the supported range a device shall adapt them instead of responding with an error.

REQUEST:

- **SearchToken [tt:JobToken]**
  Specifies the search session.
- **MinResults - optional [xs:int]**
  Specifies the minimum number of results that should be returned. If the total number of results is lower than MinResults in a completed search, all results should be returned.
- **MaxResults – optional [xs:int]**
  Specifies the maximum number of results to return.
- **WaitTime – optional [xs:duration]**
  Defines the maximum time to block, waiting for results.

FAULTS:

- **env:Sender - ter:InvalidArgVal - ter:InvalidToken**
  The search token is invalid.

ACCESS CLASS:

READ_MEDIA

RESPONSE:

- ResultList [tt:FindRecordingResultList]

  A structure containing the current SearchState and a list of RecordingInformation structures.

5.9 FindEvents

FindEvents starts a search session, looking for events in the *scope* (See 5.2.4) that match the search filter defined in the request. Events are recording events (see 5.2.2) and other events that are available in the track. Results from the search are acquired using the GetEventSearchResults request, specifying the search token returned from this request.

The device shall continue searching until one of the following occurs:

- The entire time range from *StartPoint* to *EndPoint* has been searched through.
- The total number of matches has been found, defined by the *MaxMatches* parameter.
- The session has been ended by a client EndSearch request.
- The session has been ended because *KeepAliveTime* since the last request related to this session has expired.

Results shall be ordered by time, ascending in case of forward search, or descending in case of backward search. This operation is mandatory to support for a device implementing the recording search service. Although the values of property events refer to the forward direction, they shall be reported identically in reverse search mode.

For the KeepAliveTime a device shall support at least values up to ten seconds. A device may adapt larger values.

REQUEST:

- **StartPoint [xs:dateTime]**
  The point of time where the search will start.
- **EndPoint – optional [xs:dateTime]**
  The point of time where the search will stop. This can be a time before the StartPoint, in which case the search is performed backwards in time. If EndPoint is omitted, search will go forward from the StartPoint.
- **Scope [tt:SearchScope]**
  Defines the dataset to consider for this search.
- **SearchFilter [tt:EventFilter]**
  Contains the topic and message filter needed to define what events to search for.
- **IncludeStartState [xs:boolean]**
  By setting the IncludeStartState to true, the client indicates that virtual events at the time of StartPoint should be returned to represent the state in the recording. In case of a backward search, virtual events at the time of EndPoint and StartPoint should be returned. Support for virtual events is mandatory for recording events. Support for additional virtual events is signalled via the GeneralStartEvents capability.
- **MaxMatches – optional [xs:int]**
  The search ends after MaxMatches.
- **KeepAliveTime [xs:duration]**
  The session timeout after each request concerning this session.

RESPONSE:

- **SearchToken [tt:JobToken]**
  Identifies the search session created by this request.

FAULTS:

- **env:Receiver - ter:Action - ter:ResourceProblem**
  Device is unable to create a new search session.
- **env:Sender - ter:InvalidArgVal - ter:InvalidFilterFault**
  Provided Search filter expression was not understood or supported by the device.

ACCESS CLASS:

READ_MEDIA

5.10 GetEventSearchResults

GetEventSearchResults acquires the results from a recording event search session previously initiated by a FindEvents operation. The response shall not include results already returned in previous requests for the same session. If *MaxResults* is specified, the response shall not contain more than *MaxResults* results.

GetEventSearchResults shall block until:

- *MaxResults* results are available for the response if *MaxResults* is specified.
- *MinResults* results are available for the response if *MinResults* is specified.
- *WaitTime* has expired.
- Search is completed or stopped.

This operation is mandatory to support for a device implementing the recording search service. If any of the specified parameters MinResults and WaitTime exceed the supported range a device shall adapt them instead of responding with an error.

REQUEST:

- **SearchToken [tt:JobToken]**
  Specifies the search session.
- **MinResults - optional [xs:int]**
  Specifies the minimum number of results that should be returned. If the total number of remaining events is lower than MinResults in a completed search, all remaining events should be returned.
- **MaxResults – optional [xs:int]**
  Specifies the maximum number of results to return.
- **WaitTime – optional [xs:duration]**
  Defines the maximum time to block, waiting for results.

RESPONSE:

- **ResultList [tt: FindEventResultList]**
  A structure containing:

FAULTS:

- **env:Sender - ter:InvalidArgVal - ter:InvalidToken**
  The search token is invalid.

ACCESS CLASS:

READ_MEDIA

5.11 FindPTZPosition

FindPTZPosition starts a search session, looking for ptz positions in the *scope* (See 5.2.4) that match the search filter defined in the request. Results from the search are acquired using the GetPTZPositionSearchResults request, specifying the search token returned from this request.

The device shall continue searching until one of the following occurs:

- The entire time range from *StartPoint* to *EndPoint* has been searched through.
- The total number of matches has been found, defined by the *MaxMatches* parameter.

- The session has been ended by a client EndSearch request.
- The session has been ended because *KeepAliveTime* since the last request related to this session has expired.

This operation is mandatory to support whenever CanContainPTZ is true for any metadata track in any recording on the device.

For the KeepAliveTime a device shall support at least values up to ten seconds. A device may adapt larger values.

A device shall only match the search criteria against PTZ status updates available between the time interval given in the search, i.e. the device shall not locate the PTZ position at the start of the search interval.

REQUEST:

- **StartPoint [xs:dateTime]**
  The point of time where the search will start.
- **EndPoint – optional [xs:dateTime]**
  The point of time where the search will stop. This can be a time before the StartPoint, in which case the search is performed backwards in time. If EndPoint is omitted, search will go forward from the StartPoint.
- **Scope [tt:SearchScope]**
  Defines the dataset to consider for this search.
- **SearchFilter [tt: PTZPositionFilter]**
  Contains the search criteria needed to define the PTZ position to search for.
- **MaxMatches – optional [xs:int]**
  The search ends after MaxMatches.
- **KeepAliveTime [xs:duration]**
  The session timeout after each request concerning this session.

RESPONSE:

- **SearchToken [tt:JobToken]**
  Identifies the search session created by this request.

FAULTS:

- **env:Receiver - ter:Action - ter:ResourceProblem**
  Device is unable to create a new search session.
- **env:Receiver - ter:ActionNotSupported - ter:NotImplemented**
  This optional method is not implemented.

ACCESS CLASS:

READ_MEDIA

5.12 GetPTZPositionSearchResults

GetPTZPositionSearchResults acquires the results from a PTZ position search session previously initiated by a FindPTZPosition operation. The response shall not include results already returned in previous requests for the same session. If *MaxResults* is specified, the response shall not contain more than *MaxResults* results.

GetPTZPositionSearchResults shall block until:

- *MaxResults* results are available for the response if *MaxResults* is specified.

- *MinResults* results are available for the response if *MinResults* is specified.
- *WaitTime* has expired.
- Search is completed or stopped.

This operation is mandatory to support whenever CanContainPTZ is true for any metadata track in any recording on the device. If any of the specified parameters MinResults and WaitTime exceed the supported range a device shall adapt them instead of responding with an error.

REQUEST:

- **SearchToken [tt:JobToken]**
  Specifies the search session.
- **MinResults - optional [xs:int]**
  Specifies the minimum number of results that should be returned. If the total number of results is lower than MinResults in a completed search, all results should be returned.
- **MaxResults – optional [xs:int]**
  Specifies the maximum number of results to return.
- **WaitTime – optional [xs:duration]**
  Defines the maximum time to block, waiting for results.

RESPONSE:

- **ResultList [tt:FindPTZPositionResultList]**
  A structure containing:

FAULTS:

- **env:Sender - ter:InvalidArgVal - ter:InvalidToken**
  The search token is invalid.

ACCESS CLASS:

READ_MEDIA

5.13 FindMetadata

FindMetadata starts a search session, looking for metadata in the scope (See 5.2.4) that matches the search filter defined in the request. Results from the search are acquired using the GetMetadataSearchResults request, specifying the search token returned from this request.

The device shall continue searching until one of the following occurs:

- The entire time range from *StartPoint* to *EndPoint* has been searched through.
- The total number of matches has been found, defined by the *MaxMatches* parameter.
- The session has been ended by a client EndSearch request.
- The session has been ended because *KeepAliveTime* since the last request related to this session has expired.

This operation is mandatory to support if the MetaDataSearch capability is set to true in the SearchCapabilities structure returned by the GetCapabilities command in the Device service.

For the KeepAliveTime a device shall support at least values up to ten seconds. A device may adapt larger values.

REQUEST:

- **StartPoint [xs:dateTime]**
  The point of time where the search will start.
- **EndPoint – optional [xs:dateTime]**
  The point of time where the search will stop. This can be a time before the StartPoint, in which case the search is performed backwards in time. If EndPoint is omitted, search will go forward from the StartPoint.
- **Scope [tt:SearchScope]**
  Defines the dataset to consider for this search.
- **SearchFilter [tt: MetadataFilter]**
  Contains the search criteria needed to define the metadata to search for.
- **MaxMatches – optional [xs:int]**
  The search ends after MaxMatches.
- **KeepAliveTime [xs:duration]**
  The session timeout after each request concerning this session.

RESPONSE:

- **SearchToken [tt:JobToken]**
  Identifies the search session created by this request.

FAULTS:

- **env:Receiver - ter:Action - ter:ResourceProblem**
  Device is unable to create a new search session.

ACCESS CLASS:

READ_MEDIA

5.14 GetMetadataSearchResults

GetMetadataSearchResults acquires the results from a recording search session previously initiated by a FindMetadata operation. The response shall not include results already returned in previous requests for the same session. If *MaxResults* is specified, the response shall not contain more than *MaxResults* results.

GetMetadataSearchResults shall block until:

- *MaxResults* results are available for the response if *MaxResults* is specified.
- *MinResults* results are available for the response if *MinResults* is specified.
- *WaitTime* has expired.
- Search is completed or stopped.

This operation is mandatory to support if the MetaDataSearch capability is set to true in the SearchCapabilities structure returned by the GetCapabilities command in the Device service. If any of the specified parameters MinResults and WaitTime exceed the supported range a device shall adapt them instead of responding with an error.

REQUEST:

- **SearchToken [tt:JobToken]**
  Specifies the search session.

- **MinResults - optional [xs:int]**
  Specifies the minimum number of results that should be returned. If the total number of results is lower than MinResults in a completed search, all results should be returned.
- **MaxResults – optional [xs:int]**
  Specifies the maximum number of results to return.
- **WaitTime – optional [xs:duration]**
  Defines the maximum time to block, waiting for results.

RESPONSE:

- **ResultList [tt: FindMetadataResultList]**
  A structure containing:

FAULTS:

- **env:Sender - ter:InvalidArgVal - ter:InvalidToken**
  The search token is invalid.

ACCESS CLASS:

READ_MEDIA

5.15 SearchImageByNL

SearchImageByNL starts a search session, looking for video records based on a natural language description of the content, such as "a person wearing a red hat" or "a blue car". Results from the search are acquired using the GetNLSearchResults request, specifying the search token returned from this request.

The device shall continue searching until one of the following occurs:

- The entire time range from *StartPoint* to *EndPoint* has been searched through.
- The total number of matches has been found, defined by the *MaxMatches* parameter.
- The session has been ended by a client EndSearch request.
- The session has been ended because *KeepAliveTime* since the last request related to this session has expired.

REQUEST:

- **StartPoint [xs:dateTime]**
  The point of time where the search will start.
- **EndPoint - optional [xs:dateTime]**
  The point of time where the search will stop. If EndPoint is omitted, the search will continue up to the latest recorded data available on the device.
- **RecordingToken - optional [tt:RecordingReference]**
  Token for the recording to search.
- **Text [xs:string]**
  Natural language description for the search.
- **Similarity - optional [xs:float]**
  It represents the similarity between two vectors, which is used to measure the similarity of the directions of the vectors. The closer the value is to 1, the higher the similarity; the closer the value is to 0, the lower the similarity.
- **MaxMatches - optional [xs:int]**
  The search ends after MaxMatches.

- **KeepAliveTime [xs:duration]**
  The session timeout after each request concerning this session.

RESPONSE:

- **SearchToken [tt:JobToken]**
  Identifies the search session created by this request.

FAULTS:

- **env:Receiver - ter:Action - ter:ResourceProblem**
  Device is unable to create a new search session.

ACCESS CLASS:

READ_MEDIA

5.16 GetNLSearchResults

GetNLSearchResults acquires the results from a natural language search session previously initiated by a SearchImageByNL operation. The response shall not include results already returned in previous requests for the same session.

REQUEST:

- **SearchToken [tt:JobToken]**
  Specifies the search session.
- **MinResults - optional [xs:int]**
  Specifies the minimum number of results that should be returned.
- **MaxResults – optional [xs:int]**
  Specifies the maximum number of results to return.
- **WaitTime – optional [xs:duration]**
  Defines the maximum time to block, waiting for results.

RESPONSE:

- **ResultList [tt:FindNLSearchResultList]**
  A structure containing the search results.

FAULTS:

- **env:Sender - ter:InvalidArgVal - ter:InvalidToken**
  The search token is invalid.

ACCESS CLASS:

READ_MEDIA

5.17 SearchImageByImage

SearchImageByImage starts a search session, looking for video records based on a provided image. Results from the search are acquired using the GetImageSearchResults request, specifying the search token returned from this request.

The device shall continue searching until one of the following occurs:

- The entire time range from *StartPoint* to *EndPoint* has been searched through.
- The total number of matches has been found, defined by the *MaxMatches* parameter.

- The session has been ended by a client EndSearch request.
- The session has been ended because *KeepAliveTime* since the last request related to this session has expired.

REQUEST:

- **StartPoint [xs:dateTime]**
  The point of time where the search will start.
- **EndPoint - optional [xs:dateTime]**
  The point of time where the search will stop. If EndPoint is omitted, the search will continue up to the latest recorded data available on the device.
- **RecordingToken - optional [tt:RecordingReference]**
  Token for the recording to search.
- **TargetImageURI - optional [xs:anyURI]**
  Exactly one of TargetImageURI or TargetImageData shall be provided. If both are provided, the device shall use TargetImageData. If neither is provided, the device shall respond with an InvalidArgVal fault. The URI of the detected target object image stored in the Target Image repository (LocalStorage format).
- **TargetImageData - optional [xs:base64Binary]**
  Exactly one of TargetImageURI or TargetImageData shall be provided. If both are provided, the device shall use TargetImageData. If neither is provided, the device shall respond with an InvalidArgVal fault. Base64-encoded target object image used for visual search.
- **MaxMatches - optional [xs:int]**
  The search ends after MaxMatches.
- **KeepAliveTime [xs:duration]**
  The session timeout after each request concerning this session.

RESPONSE:

- **SearchToken [tt:JobToken]**
  Identifies the search session created by this request.

FAULTS:

- **env:Receiver - ter:Action - ter:ResourceProblem**
  Device is unable to create a new search session.

ACCESS CLASS:

READ_MEDIA

5.18 GetImageSearchResults

GetImageSearchResults acquires the results from an image-based search session previously initiated by a SearchImageByImage operation. The response shall not include results already returned in previous requests for the same session.

REQUEST:

- **SearchToken [tt:JobToken]**
  Specifies the search session.
- **MinResults - optional [xs:int]**
  Specifies the minimum number of results that should be returned.
- **MaxResults – optional [xs:int]**
  Specifies the maximum number of results to return.

- **WaitTime – optional [xs:duration]**
  Defines the maximum time to block, waiting for results.

RESPONSE:

- **ResultList [tt:FindObjectImageResultList]**
  A structure containing the search results.

FAULTS:

- **env:Sender - ter:InvalidArgVal - ter:InvalidToken**
  The search token is invalid.

ACCESS CLASS:

READ_MEDIA

5.19 EndSearch

EndSearch stops an ongoing search session, causing any blocking result request to return and the *SearchToken* to become invalid. If the search was interrupted before completion, the point in time that the search had reached shall be returned. If the search had not yet begun, the *StartPoint* shall be returned. Note that an error message will occur if the search session has been already completed before this request. If the search was completed the original *EndPoint* supplied by the Find operation shall be returned. When issuing EndSearch on a FindRecordings request the *EndPoint* is undefined and shall not be used since the FindRecordings request doesn't have StartPoint/EndPoint. This operation is mandatory to support for a device implementing the recording search service.

REQUEST:

- **SearchToken [tt:Jobtoken]**
  Specifies the search session.

RESPONSE:

- **EndPoint [xs:dateTime]**
  The point in time where the search was at when ended.

FAULTS:

- **env:Sender - ter:InvalidArgVal - ter:InvalidToken**
  The search token is invalid.

ACCESS CLASS:

READ_MEDIA

5.20 GetServiceCapabilities

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

REQUEST:

This is an empty message.

RESPONSE:

- **Capabilities [tse:Capabilities]**
  Contains the requested service capabilities using a hierarchical XML capability structure.

ACCESS CLASS:

PRE_AUTH

The following capabilities are available:

MetadataSearch Indication if the device supports generic search of recorded metadata,

GeneralStartEvents Indicates support for general virtual property events in the FindEvents method

NLSearch Indicates if the device supports natural language based search for recorded video

ImageSearch Indicates if the device supports image based search for recorded video

5.21 Recording Event Descriptions

A device shall generate the following events with the corresponding event message descriptions. A device supporting the recording search service shall record these notification messages so that clients can use FindEvents to search for these messages. All recording events that are generated by the device and inserted into the recording history shall have a root topic of tns1:RecordingHistory.

The time of all recording events shall specify the actual time relating to recording regardless of the sending time of the event.

```
Topic: tns1:RecordingHistory/Recording/State
<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItemDescription Name="RecordingToken" Type="tt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="IsRecording" Type="xs:boolean"/>
  </tt:Data>
</tt:MessageDescription>
```

This message is sent whenever a client starts or stops recording for a specific recording. At start recording, IsRecording shall be set to true. At stop recording, IsRecording shall be set to false.

```
Topic: tns1:RecordingHistory/Track/State
<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItemDescription Name="RecordingToken" Type="tt:ReferenceToken"/>
    <tt:SimpleItemDescription Name="Track" Type="tt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="IsDataPresent" Type="xs:boolean"/>
  </tt:Data>
</tt:MessageDescription>
```

This message signals when the data for a track is present. When the data becomes present, a message with IsDataPresent set to true shall be sent. When the data becomes unavailable, The message with IsDataPresent set to false shall be sent.

A device MAY generate the following events. If the device supports these events, it shall always automatically record these notification messages so that clients can always use FindEvent for these messages.

Topic: tns1:RecordingHistory/Track/VideoParameters

```
<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItemDescription Name="Recording" Type="tt:ReferenceToken"/>
    <tt:SimpleItemDescription Name="Track" Type="tt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="VideoEncoding"    Type="tt:VideoEncoding"/>
    <tt:SimpleItemDescription Name="VideoWidth" Type="xs:int"/>
    <tt:SimpleItemDescription Name="VideoHeight" Type="xs:int"/>
    <tt:ElementItemDescription Name="VideoRateControl"    Type="tt:VideoRateControl"/>
```

```
  </tt:Data>
</tt:MessageDescription>
```

Topic: tns1:RecordingHistory/Track/AudioParameters

```
<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItemDescription Name="Recording" Type="tt:ReferenceToken"/>
    <tt:SimpleItemDescription Name="Track" Type="tt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="AudioEncoding"    Type="tt:AudioEncoding"/>
    <tt:SimpleItemDescription Name="AudioSampleRate" Type="xs:int"/>
    <tt:SimpleItemDescription Name="AudioBitrate" Type="xs:int"/>
  </tt:Data>
</tt:MessageDescription>
```

The device shall send either message (depending on the track data type) whenever any of these properties change.

5.22 Xpath dialect

This section defines the [W3C XPATH 1.0] dialect that a device that realises the search service shall implement to parse the Xpath strings that are passed to the methods of the search service.

```
Dialect=http://www.onvif.org/ver10/tse/searchFilter
[1] Expression ::= BoolExpr | Expression 'and' Expression
    | Expression 'or' Expression | '('Expression')' | 'not''('Expression')'
[2] BoolExpr ::= 'boolean''('PathExpr')' | 'contains''(' ElementPath ',' '"' String '"' ')'
[3] PathExpr ::= '//SimpleItem' NodeTest | '//ElementItem' NodeTest |  ElementTest
[4] NodeTest ::= '['AttrExpr']'
[5] AttrExpr ::= NameComp | ValueComp | AttrExpr 'and' AttrExpr |
    AttrExpr 'or' AttrExpr | 'not''(AttrExpr')'
[6] NameComp ::= NameAttr'='"'String'"'
[7] ValueComp ::= ValueAttr Operator '"'String'"'
[8] Operator ::= '=' | '!=' | '<' | '<=' | '>' | '>='
[9] NameAttr ::= '@Name'
[10] ValueAttr ::='@Value'
[11] ElementTest ::= '/' ElementPath '['NodeComp']'
[12] ElementPath ::= ElementName ElementName*
[13] ElementName ::= '/' String
[14] NodeComp ::= NodeName Operator '"' String '"'
[15] NodeName ::= '@' String | String
```

Example of an Xpath expression used to find recordings from the basement where there is at least one track containing video:

```
boolean(//Source[Location = “Basement”]) and boolean(//Track[TrackType = “Video”])
```

Annex A.
Deprecated Interfaces

A.1 Method for returning search status

The definition and interface for the returning search status have been deprecated with release 16.06. The following interfaces have been removed from the specification:

- GetSearchState

The definition is available via the link <https://www.onvif.org/specs/srv/rsrch/ONVIF-RecordingSearch-Service-Spec-v1606.pdf>.

Annex B.
Revision History

Rev.	Date	Editor	Changes
2.1	Jul-2011	Hans Busch	Split from Core 2.0
2.1.1	Jan-2012	Hans Busch	Change Requests 325, 327, 328, 435, 535
2.2	Apr-2012	Hans Busch	Change Requests 601, 608, 614, 616, 621, 671, 672
2.2.1	Sep-2012	Hans Busch	Change Requests 708, 765, 766, 778, 780, 781, 827, 837
2.3	May-2013	Michio Hirai	Change Request 884
2.4	Aug-2013	Michio Hirai	Change Request 1090, 1147, 1152
2.4.1	Dec-2013	Michio Hirai	Change Request 1190, 1208
2.4.2	Jun-2014	Michio Hirai	Change Request 1229
2.5	Dec-2014	Michio Hirai	Change Request 1457
2.6	Jun-2015	Michio Hirai	Change Request 1651
2.6.1	Dec-2015	Hiroyuki Sano	Change Request 1720
16.06	Jun-2016	Hiroyuki Sano	Change Request 1695, 1763
17.06	Jun-2017	Stefan Andersson, Hiroyuki Sano	Update method layouts Change Request 1843, 2050, 2053, 2055, 2065
18.12	Dec-2018	Hiroyuki Sano	Change Request 2382
21.12	Dec-2021	Sergey Bogdanov	Change the "role" attribute for request access class.
22.06	June-2022	Fredrik Svensson	Change the "role" attribute for request access class.
26.06	Jun-2026	Wang Xiaomin	Add natural language and image-based search support.