Recommendation import

Recommendation XML Import Interface Specification

Version 4.5.1

Status: Proposed

Filename	Import-Recommendation-Interface-v4.5.doc
Date	14-Sept-2016
Author(s)	christophe.sanchez@nagra.com
Client/Project	NagraMedia CMS
Owner	Christophe Sanchez

CH-1033 Cheseaux, Switzerland.
Tel: +41 21 7320311 Fax: +41 21 7320100
www.nagra.com

Contents

Contents

Preface

Audience

Nagravision product and engineering departments are the Internal attended audience.

Third party engineering departments for integration purpose are the External attended audience, and are called data provider within the document.

Document Structure

This document describes the basis of the Nagravision import interfaces that will be used for Recommendation projects.

Document Organization

This document is organized as follows:

Chapter 1: this chapter provides an overview of the import and explains the FTP mechanism used by the CMS agent tool to import automatically the XML files provided.

Chapter 2: this chapter describes the import XML file format.

Chapter 3: this chapter describes the error XML file format.

Chapter 4: Some example of import file.

Conventions Used

When the document references an XML element within the text, this element is written in another font and is placed between the XML tag delimiters like this: `<VodItem>`.

Conventions used inside the XML file are described in the §2.

Note

This document references the CMS application (GUI, import tool, database…). The name CMS is the name of the Nagravision CMS main application. The term CMS could be replaced by Nagravision CMS in this document context.

Document and Schema versionning convention

It is defined that the versionning of the Import Nagravision VOD Interface is based on three parts incremented as follow:

- The first part is incremented when a not-retrocompatible changes is performed (for instance when the root tag is changed from ScheduleProvider to NagravisionData)
- The second part is incremented when the changes impact the Schema (for instance add a new tag)

- The third part is incremented when the changes impact only the documentation (for instance a correction in an explanation or in the typography)

The Schema file has only a two parts version number and the document has the three parts.

Note

This chapter is added in the XSD too, in a comment.

List of Open Points

No table of figures entries found.

Related Documents

[1] ISO-8601 date format definition
ISO

[2] Import-Recommendation-Interface Schema
Import-Recommendation-Interface-v4.5.xsd

Document History

Change logs
4.2.0 / 2012.01.24 / Christophe Sanchez ▪ Initial version
4.4.0 / 2012.12.12 / Wendy Molyneaux ▪ Change separator in recommendation ID references from « -» to « / ». ▪ Public ID max length increated from 32 to 50 characters. ▪ Add details of usage rules.
4.4.0 / 2012.12.18 /Gilles Andre ▪ Modified pattern for EpgElement tag.
4.5.0 / 2013.05.28 / Didier von Rotz ▪ Initial version for 4.5
4.5.1 / 2013.06.13 / Léonard Porta ▪ Updated regular expression for public IDs

1. Overview

The goal of this interface is get recommendation information from a recommendation engin.

The interface proposed by Nagravision is an independent interface to exchange data with third party's systems.

The CMS import tool for this interface will be configured as an agent that will check for incoming files on a regular basis. The file will be received with FTP. The CMS agent is an ftp client, with the ftp server being on the data provider system or eventually a third party's system.

1.1 FTP import mechanism

The import software will be deployed as an agent that will run on a regular basis. When launched, the agent will retrieve the list of files to be processed by examining a specific configured directory. It will process the files in an order according to case sensitive alpha-numeric sorting of the file names. Therefore to ensure CMS processes files in the same order that they were generated it is proposed that the beginning of the file name contains the date and time of the file generation in a numeric format, year first.

In order to prevent the file access errors that would occur if the CMS agent were to try to process a file while it is still being generated or copied by the data provider, the data provider will perform a two stage provisioning operation. It will first generate or copy the file into a working directory and/or with a file extension that is different from that the CMS agent examines. Then secondly it will change directory and/or change file extension to correspond with what the CMS agent examines. If different directories are used, it is essential that they are on the same physical disk and file system so that the operating system can perform the change in an atomic action without provoking an extra data copy. In this way the CMS agent can only "see" the files once they are complete.

The CMS agent will make a local copy of the source file from the data provider's ftp server. Once this copy is done, it modifies the source file name adding the suffix ".progress" to show this file being processed.

When the CMS server has processed the file, it returns an acknowledgement to the content provider by renaming and/or changing the directory of the source file. The ".progress" suffix is removed and the suffix ".success" added if the import was successful, or ".failed" added when errors are detected during the import. Optionally the file may be moved to a different directory, with separate directories for success and failure. These directories must be on the same disk as the input directory. When the file is moved to a different directory, it is optional to add the ".success" / ".failed" suffix, so the file ends up with its original name.

In the case of failure an error file with the same name as the source file and an ".error" extension will be transmitted as described in section 3. In addition an alarm will be raised.

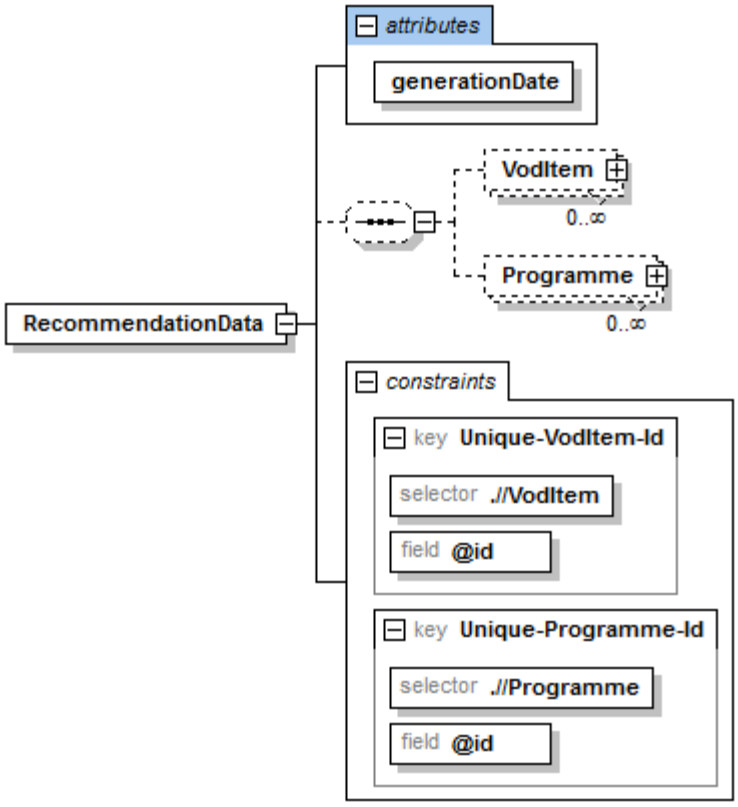
It is assumed that the data provider will take care of deleting the processed files once it has taken note of the acknowledgement.

2. Nagravision XML file format

2.1 File encoding

The character encoding of the source file must corresponding to the "encoding" attribute of the first line of the xml file. Nagravision advises to use UTF8 encoding in order to easily manage ASCII and non-ASCII texts in a uniform way.

2.2 Root element



`<RecommendationData>` is the root element. It may contain any number of `<VodItem>` or `<Programme>` child elements. The data provider can choose how he organizes how many elements he puts in each separate xml file. He may put one element per file, or group all data in a single file. When deciding upon this organization it needs to be taken into account that if an error is found while processing a file, none of the data changes implied by that file will be applied to the CMS database. Therefore for optimal performance Nagravision advises providing XML files that contain only new items or updates, and grouping about 100 elements per input file.

Attribute details

Name	Description
`generationDate`	Date and time of file generation.

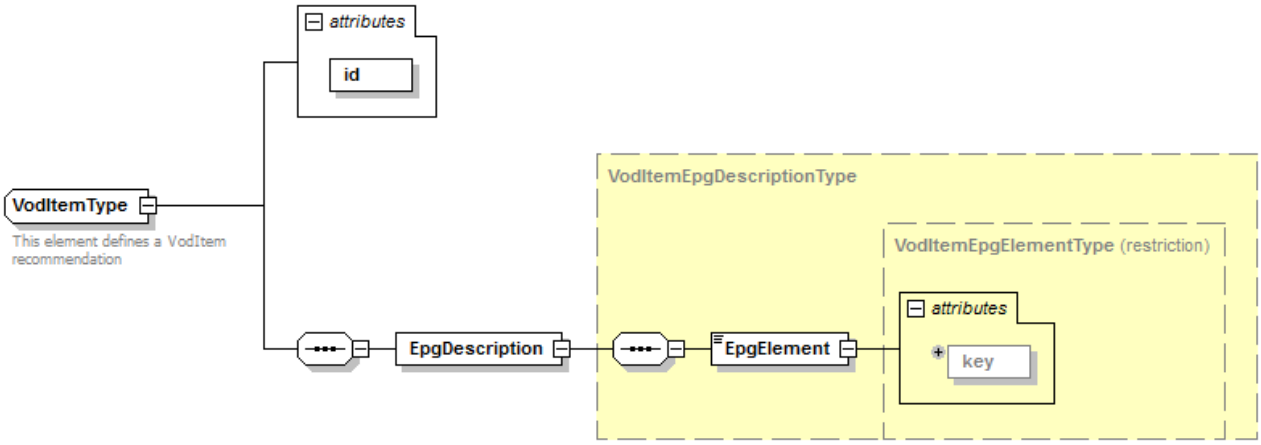
The `generationDate` attribute must be present and syntactically correct but its value is not further processed by the current CMS import tool. Nagravision recommends that the data provider supplies the timestamp of the file's generation for documentation purposes.

Note

A key constraint is defined to insure the unicity of the Id attributes of the <VodItem> and the <Programme> elements within the entire XML document.

2.3 VodItem element

The `<VodItem>` element within `<RecommendationData>` is defined by the type VodItemType:

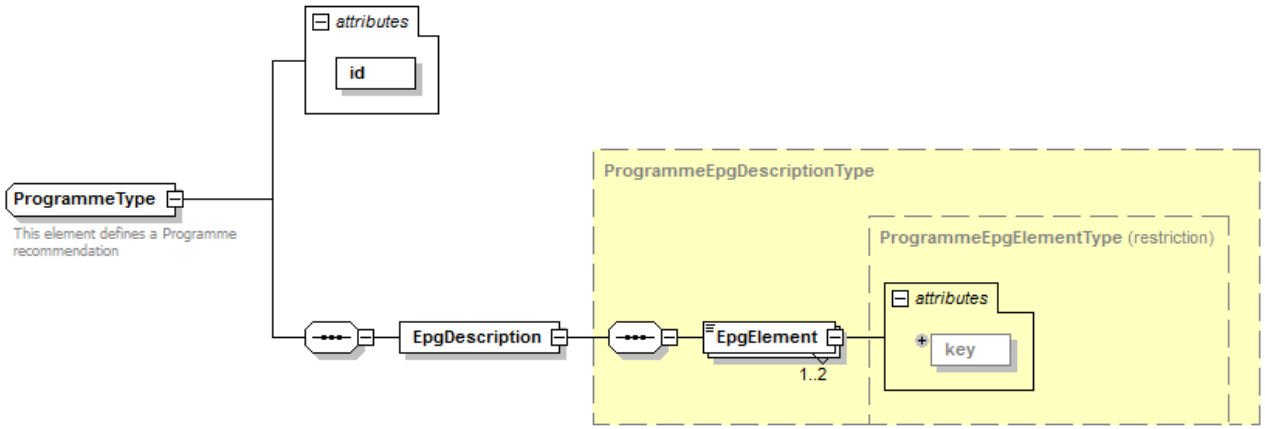


Attribute details

Name	Description
`Id`	Unique id of the VOD item to put recommendation on, see §2.7. If no such VOD item exists in the CMS database, the CMS import will report a warning and ignore the element. The remainder of the file will still be processed.

2.4 Programme element

The <Programme> element within <RecommendationData> is defined by the type ProgrammeType:



Attribute details

Name	Description
id	Unique id of the Programme to put recommendation on, see §2.7. If no such programme exists in the CMS database, the CMS import will report a warning and ignore the element. The remainder of the file will still be processed.

2.5 EpgDescription element

The <EpgDescription> elements within <VodItem> and <Programme> provides a container for supplying a set of generically defined data values. Each value is defined as a "keyword – value" pair in a <EpgElement> child element.

2.6 EpgElement element

The <EpgElement> elements within <EpgDescription> provide the value of an EPG field of the object in which it is embedded.

The list of allowed field names (key attribute) and the format of their values is defined below.

2.6.1 Attribute "key" details

Parent	Description
VodItem	Case sensitive name of the EPG field. The only allowed key is "RecommendedVodItemIds"
Programme	Case sensitive name of the EPG field. The list of allowed keys are "RecommendedVodItemIds" and "RecommendedProgrammeIds"

2.6.2 Values details

Field name	Value Description	Example
RecommendedVodItemIds	The VodItemId corresponds to the public id of the recommended VOD item. (see §2.7). No validation against the real presence of this VodItem in the CMS DB will be done.	LYS12345/10
RecommendedProgrammeIds	The ProgrammeId corresponds to the public id of the recommended programme (see §2.7). No validation against the real presence of this programme in the CMS DB will be done.	LYS1/3;7894/0

The values of both fields have the same structure: Semi-colon separated list (without whitespace) of the following pattern:

<Id><separator><Rank>

<Id> The unique identifier of a Voditem or Programme in the CMS database.

<separator> is a single slash ("/").Prior to CMS4.4 the separator was defined as a dash, which was not allowed in an ID. Starting from CMS4.4 dashes are allowed in IDs and therefore the separator is changed to slash, which is not an allowed character for ID's.

<Rank> is an unsigned integer that is mandatory.
A high rank value means that the recommended VodItem is highly recommended.
If the recommendation engine doesn't support the ranking feature, this value must be set to '0'.

Values are matched against the following regular expression:

```
([a-zA-Z0-9_]+[a-zA-Z0-9_\-]*/[0-9]+(;[a-zA-Z0-9_]+[a-zA-Z0-9_\-]*/[0-9]+)*)?
```

Notice

This pattern allows emptiness for EpgElement tag.

A particular Id should appear only once in the list. It is the responsibility of the person or system supplying the values to respect this rule. The CMS will NOT check the uniqueness of IDs within each list.

When providing an update for a voditem or programme given in a previous file, the full list of recommendations must be supplied. The specified list of recommendations will entirely replace those specified in an earlier file (or entered via the CMS GUI).

The value may be empty. This case is used to remove all the recommendations supplied in a previous file for the same Voditem or programme.

Ex:

```
<Programme id="LYS000050250">
   <EpgDescription>
      <EpgElement key="RecommendedProgrammeIds">L/2</EpgElement>
      <EpgElement key="RecommendedVodItemIds">L/3;L/4</EpgElement>
   </EpgDescription>
</Programme>
```

2.7 Technical identifier

Some elements in this schema have an `id` attribute (xsd IdType) for the following reasons:

a) To reference the element from an other element;

b) Only elements with an `id` could be updated

The `id` attribute is used to identify the item in the CMS database to be updated. The maximum possible length is 50 characters. An ID should generally be composed of printable ASCII 7-bit characters without spaces. Depending of the projects, the format and length of this ID can varied. By default the following format shall generally be used:

Only characters [a-zA-Z0-9] dash and underline are allowed.

Example: LYS_012345678-3

2.8 Date formats

The date format is based on ISO-8601 [1]. **All dates are given in GMT time**. The mandatory format is:

- SimpleType gmtdatetime: `yyyy-mm-ddThh:mm:ssZ`

The dashes, colons, T and Z characters are mandatory field separators.

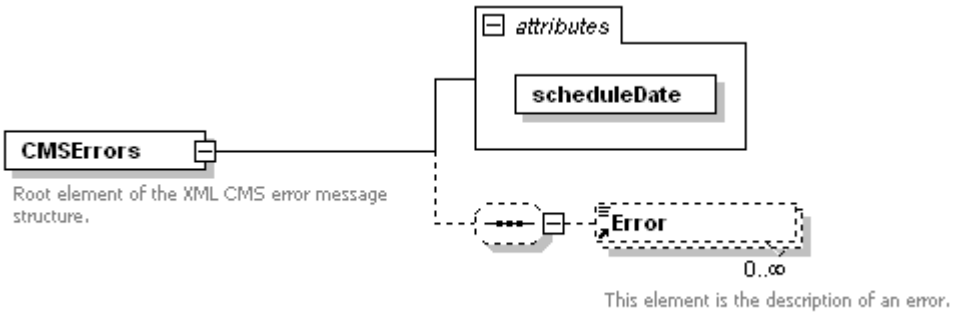
In terms of schema validation this type is a restriction of string, with the following pattern applied:

- gmtdatetime: `[0-9][0-9][0-9][0-9]-[0-1][0-9]-[0-3][0-9]T[0-2][0-9]:[0-6][0-9]:[0-6][0-9]Z`

3. Error file format

In the case where the CMS import process detects an error causing it to reject the input xml file an error file with the same name as the source file and an ".error" extension will be generated and written in the acknowledgement failure directory.

The error file is an XML file with the following structure



<CMSErrors> attributes details		
Name	**Type**	**Description and typical value**
scheduleDate	gmtdatetime Mandatory	See§ 2.8 Provides a reference date for the error XML file. Will be the date and time that the CMS generates the file. Eg "2011-08-04T08:15:58Z"

The child <Error> elements have type "xs:string", with undefined maximum length. Multiple <Error> elements may be supplied, one for each error detected.

3.1 Error file sample

In case of problems, the response should include a description in an xml format (content-type: text/xml).

```
<?xml version="1.0" encoding="UTF-8"?>
<CMSErrors scheduleDate="2004-07-01T22:00:00Z">
<Error>Error description 1</Error>
<Error>Error description 2</Error>
<Error>Error description 3</Error>
</CMSErrors>
```

3.2 Error file xsd schema

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
   <xs:element name="CMSErrors">
      <xs:annotation>
         <xs:documentation>Root element of the XML CMS error message structure.
         </xs:documentation>
      </xs:annotation>
      <xs:complexType>
         <xs:sequence minOccurs="0">
            <xs:element ref="Error" minOccurs="0" maxOccurs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="scheduleDate" type="gmtdatetime" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="Error">
      <xs:annotation>
         <xs:documentation>This element is the description of an error.
         </xs:documentation>
      </xs:annotation>
      <xs:complexType>
         <xs:simpleContent>
            <xs:extension base="xs:string"/>
         </xs:simpleContent>
      </xs:complexType>
   </xs:element>
   <xs:simpleType name="gmtdatetime">
      <xs:annotation>
         <xs:documentation>A ISO 8601 compatible gmt datetime
                           Format : yyyy-mm-ddThh:mm:ssZ
         </xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
         <xs:pattern value="[0-9][0-9][0-9][0-9]-[0-1][0-9]-[0-3][0-9]T[0-2][0-
9]:[0-6][0-9]:[0-6][0-9]Z"/>
      </xs:restriction>
   </xs:simpleType>
</xs:schema>
```

4. XML Samples

4.1 Import file Sample

```
<RecommendationData generationDate="2012-05-25T10:45:00Z"
xsi:noNamespaceSchemaLocation="Import-Recommendation-Interface-v4.4.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <VodItem id="LYS2">
      <EpgDescription>
         <EpgElement key="RecommendedVodItemIds">LYS1234/2</EpgElement>
      </EpgDescription>
   </VodItem>
   <VodItem id="LYS3">
      <EpgDescription>
         <EpgElement key="RecommendedVodItemIds">LYS12340/3;LYS12340/4</EpgElement>
      </EpgDescription>
   </VodItem>
   <Programme id="0">
      <EpgDescription>
         <EpgElement key="RecommendedVodItemIds">LYS123400/26</EpgElement>
         <EpgElement key="RecommendedProgrammeIds">LYS12340/123;LYS12340/56</EpgElement>
      </EpgDescription>
   </Programme>
   <Programme id="1">
      <EpgDescription>
         <EpgElement key="RecommendedVodItemIds">LYS123400/34</EpgElement>
      </EpgDescription>
   </Programme>
</RecommendationData>
```

End of document