Publications Office

Framework Contract No 10373 CELLAR

Maintenance and Evolution of the CELLAR External End User Manual

Subject	Cellar's External End User manual
Version	13.01
Release Date	14/08/2015
Filename	CEM-EEU - External End User Manual-v13.01.doc
Document Reference	CEM-EEU-External End User manual

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS AND ACRONYMS

ABBREVIATIONS AND ACRONYMS	
Abbreviation	**Meaning**
CURIE	Compact URI
cURL	Client URL Request Library
FRBR	Functional Requirements for Bibliographic Records
JSON	JavaScript Object Notation
NAL	Named Authority List
OWL	Web Ontology Language
RDF	Resource Description Framework
SKOS	Simple Knowledge Organization System
SPARQL	SPARQL Protocol and RDF Query Language
URI	Uniform Resource Identifier
UUID	Universally Unique Identifier
WEMI	Work, Expression, Manifestation and Item
XML	Extensible Markup Language

Table 1 – Abbreviations and Acronyms

DEFINITIONS	
Term	**Meaning**
ISO_639-3	Codes for the representation of names of languages
UUID	Universally Unique Identifier: identifier standard used in software construction, standardized by the Open Software Foundation (OSF)

Table 2 – Definitions

1 INTRODUCTION

1.1 PURPOSE OF THE DOCUMENT

The purpose of this document is to provide the CELLAR end-user with a structured, non-technical, easy-to-read user manual. Actually, the existing documents are too technical to be provided to the end-user as valid alternatives.

1.2 INTENDED AUDIENCE

This document is intended for all the CELLAR end-users.

1.3 STRUCTURE OF THE DOCUMENT

The document is organized as follows:

- **Chapter 1**: the present Introduction;
- **Chapter 2**: a presentation of the main concepts on which the CELLAR is built upon;
- **Chapter 3**: a full description of CELLAR's available services including some usage scenarios.

2 MAIN CONCEPTS

Here follows a description of the main concepts on which the CELLAR data model is built upon:

- Functional Requirements for Bibliographic Records (FRBR) – paragraph 2.1
- Types of notices – paragraph 2.2
- Content streams – paragraph 2.3
- NALs – paragraph 2.4
- EUROVOC – paragraph 2.5
- Resource URI – paragraph 2.6

2.1 FUNCTIONAL REQUIREMENTS FOR BIBLIOGRAPHIC RECORDS (FRBR)

Functional Requirements for Bibliographic Records (*FRBR*) is a conceptual entity-relationship model developed by the *International Federation of Library Associations and Institutions* (*IFLA*) that relates user tasks of retrieval and access in online library catalogues and bibliographic databases from a user's perspective.

The FRBR comprises 3 groups of entities.

The group 1 entities are the *Work*, *Expression*, *Manifestation*, and *Item* (*WEMI*): they represent the products of intellectual or artistic endeavour, and are the foundation of the FRBR model.

Here follows a description of each:

- the *Work* is generally defined as a *distinct intellectual or artistic creation*. Example: Beethoven's Ninth Symphony apart from all ways of expressing it is a work
- the *Expression* is *the specific intellectual or artistic form that a work takes each time it is 'realized*'. Example: an expression of Beethoven's Ninth might be the musical score he writes down
- the *Manifestation* is *the physical embodiment of an expression of a work*. As an entity, manifestation represents all the physical objects that bear the same characteristics, in respect to both intellectual content and physical form. Example: the recording the London Philharmonic made of the Ninth in 1996 is a manifestation
- the *Item* is *a single exemplar of a manifestation*. The entity defined as item is a concrete entity. Example: each of the 1996 pressings of that 1996 recording is an item.

The group 2 entities are *Person* and *Corporate body*, responsible for the custodianship of Group 1's intellectual or artistic endeavor.

The group 3 entities are subjects of Group 1 or Group 2's intellectual endeavour, and include *Concepts*, *Objects*, *Events* and *Places*.

2.1.1 FRBR IN CELLAR'S CONTEXT

For what concerns its use in the CELLAR, the essential idea of FRBR is to present a publication at different levels of abstraction. In order to accomplish this, the CELLAR realizes the WEMI pattern through three different hierarchies, each with its own levels of abstraction.

A) Hierarchy work-expression-manifestation-content stream

The work-expression-manifestation-content stream hierarchy (see Figure 1) is composed by:

- a work, which covers the W role of the WEMI pattern. A work may embed:
- several expressions. An expression covers the E role of the WEMI pattern, and is defined as *the realization of a work in a specific language*. It may embed:
- several manifestations. A manifestation covers the M role of the WEMI pattern, and is defined as *the instantiation of a work in the language defined by the embedding expression, and in a specific format*. Finally, a manifestation may embed:
- several content streams. A content stream covers the I role of the WEMI pattern, and is defined as *the entity that physically carries the information of the manifestation*. The content stream is typically a document written in the language and format defined by the embedding manifestation.

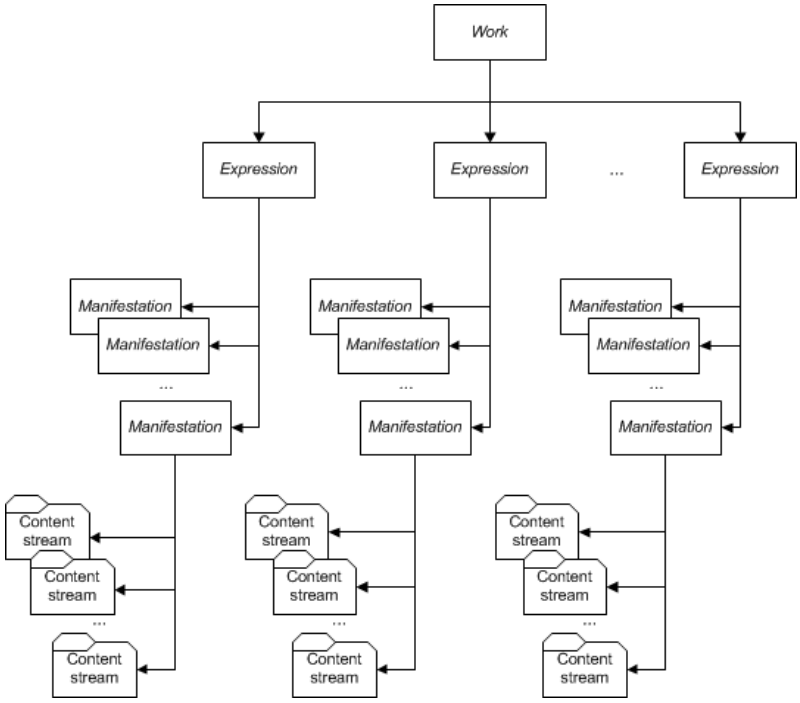

Figure 1 – the work-expression-manifestation-content stream hierarchy

The Cellar contains works from the OP's primary domains of work:

- Legislative data, currently published primarily in the EUR-Lex portal
- General publications, currently published in EU Bookshop
- Tender documents and related works (OJ-S), currently published in TeD portal
- Research documents, currently published in the CORDIS portal

The WEM model is applied consistently throughout for works from all domains. However, the abstract classes such as work are then concretized for the various domains in the Cellar's Common Data Model (CDM) by *subclassing* these abstract classes. The full set of subclasses is documented in the

CDM's wiki page available under http://www.cc.cec/wikis/display/OP/CMR+Common+Data+Model and are beyond the scope of this manual

B) Hierarchy dossier-event

The dossier-event hierarchy (see

Figure *2*) is composed by:

- a dossier, which covers the W role of the WEMI pattern. A dossier may embed:
- several events, which cover the E role of the WEMI pattern.

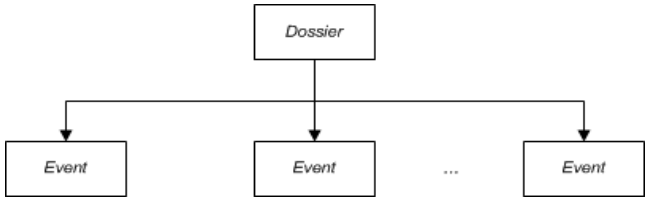

Figure 2 – the dossier-event hierarchy

As for works, dossiers can have specializations for each of the domains. At present there are such specializations for legislative procedures with and without inter-institutional codes to classify legislative procedures. There are also classifications for different types of events that can occur in a procedure.

C) Hierarchy agent

The agent hierarchy is solely composed by an agent, which covers the W role of the WEMI pattern.

These realizations of the WEMI pattern are the basis of the CELLAR's definition and data layer, that is, its *ontology*.

2.2 TYPES OF NOTICES

We present hereby the concept of notice, which can be subsequently divided into 5 types: *tree*-, *branch*-, *object*-, *identifier*- and *rdf-notice*.

For the sake of simplicity, the explanations below refer to the work-expression-manifestation-content stream hierarchy, but they can be considered valid also for the dossier-event and agent hierarchy.

2.2.1 TREE NOTICE

A *Tree notice* is an XML document including:

- the work's metadata
- all available expressions' metadata
- all available manifestations' metadata for each expression.

All metadata is decoded in the given *decoding language*, that is, the language used for notices to decode NAL and EUROVOC concepts into the specific natural language. For more information about NAL and EUROVOC concepts, please consult paragraphs 2.4 and 2.5.

For more information about how to retrieve a tree notice and its format, please see paragraph 3.1.1.

2.2.2 BRANCH NOTICE

A *Branch notice* is a content language specific XML document including:

- the work's metadata
- the metadata of the expression in the given content language
- all available manifestations' metadata for that expression.

All metadata is decoded in the given decoding language.

It is a subset of the Tree Notice.

For more information about how to retrieve a branch notice and its format, please see paragraph 3.1.2.

2.2.3 OBJECT NOTICE

An *Object notice* is a content language specific XML document with the metadata for a specific resource (work/expression/manifestation).

The metadata is decoded in the given decoding language.

It is a subset of the Tree Notice because only one object is in scope, while hierarchically dependent objects are not included (e.g. an expression, but not its manifestations).

For more information about how to retrieve an object notice and its format, please see paragraphs 3.1.3, 3.1.4 and 3.1.5.

2.2.4 IDENTIFIER NOTICE

An *Identifier notice* is an XML document containing the synonyms of a list of resource URIs.

For a definition of resource URI, please see paragraph 2.6.

For more information about how to retrieve an identifier notice and its format, please see paragraph 3.1.6.

2.2.5 RDF-OBJECT NOTICE

An *RDF-Object notice* is the RDF/XML notice format for a specific resource (work/expression/manifestation/dossier/event/agent).

For more information about how to retrieve an RDF-Object notice and its format, please see paragraph 3.1.7.

2.2.6 RDF-TREE NOTICE

An *RDF-Tree notice* is the RDF/XML notice format for the tree whose root is a specific resource (work/dossier /agent).

For more information about how to retrieve an RDF-Tree notice and its format, please see paragraph 3.1.7

2.3 CONTENT STREAMS

The *content stream* physically carries the information of the manifestation that embeds it. It realizes the *item* of the WEMI pattern (see also paragraph 2.1.1).

Typically, it is a document written in the content language and format defined by the embedding manifestation: for instance, it may represent the PDF document *Official Journal of the European Union C-318, Volume 52*, English edition.

For more information about how to retrieve a content stream, please see paragraph 3.1.9.

2.4 NALs

The *NALs (Named Authority List)* are a preloaded, not modifiable, decoded-by-language set of data meant to be used by the Cellar ontology's concepts. The NAL itself is a concept defined with the resource URI:

`http://publications.europa.eu/resource/authority/*`

where * is the NAL specific class, which currently can be one of:

```
corporate-body, country, currency, document-collection, event, fd_010, fd_013,
fd_014, fd_020, fd_030, fd_040, fd_050, fd_060, fd_070, fd_080, fd_091, fd_100,
fd_110, fd_130, fd_140, fd_160, fd_271, fd_274, fd_285, fd_290, fd_300, fd_301,
fd_305, fd_325, fd_330, fd_335, fd_340, fd_345, fd_350, fd_361, fd_365, fd_370,
fd_375, fd_380, fd_395, fd_396, fd_400, fd_401, fd_406, fd_499, fd_500, fd_501,
fd_535, fd_537, fd_555, fd_557, fd_577, fd_600, fd_601, fd_602, fd_603, fd_604,
fd_605, fd_606, fd_607, fd_608, fd_609, fd_610, fd_611, fd_612, fd_613, language,
ntu???, place, procedure, role, target-audience, treaty
```

One exception is EUROVOC, which is defined at:

`http://eurovoc.europa.eu/100141`

2.5 EUROVOC

Eurovoc is the multilingual thesaurus maintained by the Publications Office of the European Union.

It exists in all the official languages of the European Union.Eurovoc is used by:

- the European Parliament
- the Office for Official Publications of the European Union
- the national and regional parliaments in Europe
- some national government departments and European organisations.

This thesaurus serves as the basis for the domain names used in the European Union's terminology database: *Inter-Active Terminology for Europe*.

As stated in previous paragraph, the EUROVOC is one specific type of NAL.

2.6 RESOURCE URI

Each resource in the CELLAR is globally identified by a URI composed as follows:

```
http://publications.europa.eu /resource/{ps-name}/{ps-id}
```

From now on, we will refer to this URI as the *resource URI*.

Here follows a description of each part of the resource URI (paragraphs 2.6.1 and 2.6.2), with some examples depicted in paragraph 2.6.3. Finally, paragraph 2.6.4 describes the *CURIE* format.

2.6.1 {PS-NAME}

It identifies the name of the production system.

The CELLAR currently uses the following production system names: `cellar, celex, oj, com, genpub, ep, jurisprudence, dd, mtf, consolidation, eurostat, eesc, cor,`

nim, pegase, transjai, agent, uriserv, join, swd, comnat,mdr, legissum, ecli, procedure, procedure-event, eli, immc and planjo.

2.6.2 {PS-ID}

It is the resource's unique identifier, and it has a structure that depends on the value of {ps-name}.

2.6.2.1 If {ps-name} is 'cellar'

cellar is the only production system's name reserved to the CELLAR application, and its identifiers follow the following conventions:

Type	{ps-id}	Example
work dossier event agent	{work-id}	550e8400-e29b-41d4-a716-446655440000
expression	{work-id}.{expr-id}	550e8400-e29b-41d4-a716-446655440000.0001
manifestation	{work-id}.{expr-id}.{man-id}	550e8400-e29b-41d4-a716-446655440000.0001.03
content stream	{work-id}.{expr-id}.{man-id}/{cs-id}	550e8400-e29b-41d4-a716-446655440000.0001.03/DOC_1

Table 3 – Identifier's conventions for production system name *cellar*

where:

- {work-id} is a valid *Universally Unique Identifier (UUID)*
- {expr-id} is a 4-chars numeric value
- {man-id} is a 2-chars numeric value
- {cs-id} is an alphanumeric value with following pattern: DOC_x, where x is an incremental numeric value that identifies the content stream.

2.6.2.2 If {ps-name} is other than 'cellar'

For all other production system's names, the following conventions are used:

Type	{ps-id}	Example
work dossier agent	{work-id}	32006D0241
expression	{work-id}.{expr-id}	32006D0241.fra
manifestation	{work-id}.{expr-id}.{man-id}	32006D0241.fra.fmx4
content stream	{work-id}.{expr-id}.{man-id}.{cs-id}	32006D0241.fra.fmx4.L_2006088FR.01006402.xml

event	{work-id}.{event-id}	11260.12796

Table 4 – Identifier's conventions for production system names other than *cellar*

where:

- {work-id} is an alphanumeric value
- {expr-id} is a 3-chars ISO_639-3 language code. For the exhaustive list of supported ISO_639-3 codes, please refer to paragraph 4.1.
- {man-id} is an alphanumeric value identifying a file format (FORMEX, PDF, HTML, XML, etc.)
- {cs-id} is an alphanumeric value identifying the name of the content stream
- {event-id} is a numeric value.

2.6.3 EXAMPLES OF VALID RESOURCE URIS

Here follows a non-exhaustive list of examples of resource URIs that match the patterns described above:

1) The following resource URI identifies a work with ps-name of type *cellar* and the given ps-id:

   http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-01aa75ed71a1

2) The following resource URI identifies an expression – belonging to the work at point 1) – with ps-name of type *cellar* and the given ps-id:

   http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-01aa75ed71a1.0006

3) The following resource URI identifies a manifestation – belonging to the expression at point 2) - with ps-name of type *cellar* and the given ps-id:

   http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-01aa75ed71a1.0006.03

4) The following resource URI identifies a content stream – belonging to the manifestation at point 3) – with ps-name of type *cellar* and the given ps-id:

   http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-01aa75ed71a1.0006.03/DOC_1

5) The following resource URI identifies a work with ps-name of type *oj* and the given ps-id:

   http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01

6) The following resource URI identifies a work with ps-name of type *celex* and the given ps-id:

   http://publications.europa.eu/resource/celex/32014R0001

7) The following resource URI identifies an expression – belonging to the work at point 6) - with ps-name of type *celex* and the given ps-id:

   http://publications.europa.eu/resource/celex/32014R0001.FRA

8) The following resource URI identifies a manifestation – belonging to the expression at point 7) - with ps-name of type *oj* and the given ps-id:

   http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01.FRA.fmx4

9) The following resource URI identifies a content stream – belonging to the manifestation at point 8) - with `ps-name` of type *oj* and the given `ps-id`:

   http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01.FRA.fmx4.L_2014001FR.01000302.xml

10) The following resource URI identifies a work with `ps-name` of type *pegase* and the given `ps-id`:

    http://publications.europa.eu/resource/pegase/11260

11) The following resource URI identifies an event with `ps-name` of type *pegase* and the given `ps-id`:

    http://publications.europa.eu/resource/pegase/11260.12796

2.6.4 CURIE FORMAT OF A RESOURCE URI

For practical reasons, resource URIs are abbreviated onto a *CURIE (Compact URI)* format. This is done by making the production system name the alias of the system base URI.

For example, by declaring the namespace

`xmlns:celex=`http://publications.europa.eu/resource/celex/

we can abbreviate

http://publications.europa.eu/resource/celex/1234R5678

onto

`celex:1234R5678`

This CURIE format is important as it is massively used for identifying objects in Cellar's notices (for more info about Cellar's notices' format, please see paragraph 3.1).

3 AVAILABLE SERVICES

The CELLAR API allows performing different operations on the CELLAR. Such API encapsulates all the HTTP calls to the CELLAR and exposes convenience methods allowing the user to easily retrieve the requested content.

It is hereby described how to invoke services on WEMI objects, namely:

- retrieve the tree notice of a work – see paragraph 3.1.1
- retrieve the branch notice of a work – see paragraph 3.1.2
- retrieve the object notice of an object (work, expression or manifestation) – see paragraphs 3.1.3, 3.1.4 and 3.1.5.
- retrieve all the identifiers of a specific document (synonyms) – paragraph 3.1.6
- retrieve the RDF/XML formatted metadata for a given resource – paragraph 3.1.7
- retrieve content streams of a work given a specific language and format – paragraph 3.1.9

and how to invoke services on NAL/EUROVOC objects, namely:

- retrieve a dump – paragraph 3.2.1
- retrieve the supported languages – paragraph 3.2.2
- retrieve a concept scheme – paragraph 3.2.3
- retrieve the concept schemes – paragraph 3.2.4
- retrieve a concept – paragraph 3.2.5
- retrieve the concept relatives – paragraph 3.2.6
- retrieve the top concepts – paragraph 3.2.7
- retrieve the domains – paragraph 3.2.8.

The next sections explain how to use these services, each of which is described through the following sections:

- description: a short description of what the service is supposed to do
- request, where are described:
    - the URL to invoke and its type (GET or POST)
    - the URL parameters, if any. Please note that all parameters representing an HTTP URL themselves must be URL-encoded, for example: `http%3A%2F%2Fpublications.europa.eu%2Fresource%2Fauthority%2Ffd_330`

      If not specified otherwise, a parameter is always *mandatory*
    - the HTTP headers, if any
    - a list of examples of valid requests.
- response: what the response is supposed to contain, its format, and an example of it.

3.1 WEMI SERVICES

We describe hereby the available services for retrieving the information related to the WEMI objects. For simplicity, they are described for the work-expression-manifestation-content stream hierarchy, but they can be considered valid also for the dossier-event and agent hierarchy (see paragraph 2.1.1).

Dissemination service uses a global negotiation system that returns always a “303 - See other” response. The client must enable the follow-redirect option.

3.1.1 RETRIEVE THE TREE NOTICE

Description

This service allows the user to search for a complete tree notice of a given work, decoded in the given decoding language.

The returned notice will contain the work metadata, the metadata of all the expressions associated to the work, and the metadata of all the manifestations associated to the expressions.

Request

The user must fire a GET request to the following URL:

`http://publications.europa.eu/resource/{ps-name}/{ps-id}?language={dec-lang}&filter={in_notice-only}`

where:

- `{ps-name}` is a valid production system name (see also paragraph 2.6.1)
- `{ps-id}` is a valid production system id identifying a work, and compatible with its `{ps-name}` (see also paragraph 2.6.2)
- `{dec-lang}` is a 3-chars ISO_639-3 language code identifying the decoding language to use: this is the language used for decoding the NALs associated to the notice. If decoding language is not available, the default value defined in the configuration is used.
- `{in_notice-only}` is an optional boolean that indicates if the notice contains only the properties annotated with in_notice.

  Please note: no matter what the request specifies, the response notice is always the filtered one. The filter parameter will stay for a transition period due to legacy reasons.

The following HTTP headers must be set on the request:

- `Accept:application/xml;notice=tree`

Here follows some examples of valid requests using *cURL* (for a brief description about what cURL is and how to use it, please refer to paragraph 4.2):

- `curl -H "Accept:application/xml;notice=tree" "http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01?language=eng"`
- `curl -H "Accept:application/xml;notice=tree" "http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01?language=eng"`
- `curl -H "Accept:application/xml;notice=tree" "http://publications.europa.eu/resource/celex/32014R0001?language=eng"`

Please note that the 3 requests use different production system names and identifiers, but actually retrieve the s*ame* work. These 3 synonyms are related to the same cellar id.

Response

The response is an XML-formatted tree notice containing the full hierarchy of the work, here included all the expressions of the work and all the manifestations associated to the expressions.

Here follows an example of returned notice (only the relevant information is reported):

```
<NOTICE decoding="eng" type="tree">
   <WORK>
      <URI>
         <VALUE>http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-
8e20-01aa75ed71a1</VALUE>
         <IDENTIFIER>b84f49cd-750f-11e3-8e20-01aa75ed71a1</IDENTIFIER>
         <TYPE>cellar</TYPE>
      </URI>
      <SAMEAS>
         <URI>
            <VALUE>http://publications.europa.eu/resource/celex/32014R0001</VALUE>
            <IDENTIFIER>32014R0001</IDENTIFIER>
            <TYPE>celex</TYPE>
         </URI>
      </SAMEAS>
      <SAMEAS>
         <URI>

<VALUE>http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01</VALUE>
            <IDENTIFIER>JOL_2014_001_R_0001_01</IDENTIFIER>
            <TYPE>oj</TYPE>
         </URI>
      </SAMEAS>
      [...]
   </WORK>
   [...]
   <EXPRESSION>
      [content of expression 0001]
   <EXPRESSION>
   <MANIFESTATION>
      [content of manifestation 0001.01]
   <MANIFESTATION>
   <MANIFESTATION>
      [content of manifestation 0001.02]
   <MANIFESTATION>
   [...]
   <MANIFESTATION>
      [content of manifestation 0001.M]
   <MANIFESTATION>
   <EXPRESSION>
      [content of expression 0002]
   <EXPRESSION>
   <MANIFESTATION>
      [content of manifestation 0001.01]
   <MANIFESTATION>
   <MANIFESTATION>
      [content of manifestation 0002.02]
   <MANIFESTATION>
   [...]
   <MANIFESTATION>
      [content of manifestation 0002.M]
   <MANIFESTATION>
   [...]
   <EXPRESSION>
      [content of expression N]
   <EXPRESSION>
   <MANIFESTATION>
      [content of manifestation N.01]
```

```
   <MANIFESTATION>
   <MANIFESTATION>
      [content of manifestation N.02]
   <MANIFESTATION>
   [...]
   <MANIFESTATION>
      [content of manifestation N.M]
   <MANIFESTATION>
</NOTICE>
```

3.1.2 RETRIEVE THE BRANCH NOTICE

Description

This service allows the user to search for a complete branch notice of a given work, decoded in the given decoding language.

The returned notice will contain the work metadata, the metadata of the expression in the given accept language, and the metadata of all manifestations associated to the expression.

Request

The user must fire a GET request to the following URL:

`http://publications.europa.eu/resource/{ps-name}/{ps-id}?language={dec-lang}&filter={in_notice-only}`

where:

- `{ps-name}` is a valid production system name
- `{ps-id}` is a valid production system id identifying a work, and compatible with its `{ps-name}`
- `{dec-lang}` is a 3-chars ISO_639-3 language code identifying the decoding language to use: this is the language used for decoding the NALs associated to the notice. If decoding language is not available, the default value defined in the configuration is used.
- `{in_notice-only}` is an optional boolean that indicates if the notice contains only the properties annotated with in_notice.

  Please note: no matter what the request specifies, the response notice is always the filtered one. The filter parameter will stay for a transition period due to legacy reasons.

The following HTTP headers must be set on the request:

- `Accept:application/xml;notice=branch`
- `Accept-Language:{acc-lang},` where `{acc-lang}` is a 3-chars ISO_639-3 language code identifying the accept language to use: this will be used for retrieving the correct expression.

Here follows some examples of valid requests that retrieve the same object, using cURL:

- `curl -H "Accept:application/xml;notice=branch" -H "Accept-Language:fra" "http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-01aa75ed71a1?language=eng"`
- `curl -H "Accept:application/xml;notice=branch" -H "Accept-Language:fra" http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01?language=en"`
- `curl -H "Accept:application/xml;notice=branch" -H "Accept-Language:fra" "http://publications.europa.eu/resource/celex/32014R0001?language=eng"`

Response

The response is an XML-formatted branch notice containing the work, within the expression in the given accept language, and all the associated manifestations.

Here follows an example of returned notice:

```
<NOTICE decoding="eng" type="branch">
   <WORK>
      <URI>
         <VALUE> http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-
8e20-01aa75ed71a1</VALUE>
         <IDENTIFIER> b84f49cd-750f-11e3-8e20-01aa75ed71a1 </IDENTIFIER>
         <TYPE>cellar</TYPE>
      </URI>
      <SAMEAS>
         <URI>
            <VALUE>http://publications.europa.eu/resource/celex/32014R0001</VALUE>
            <IDENTIFIER>32014R0001</IDENTIFIER>
            <TYPE>celex</TYPE>
         </URI>
      </SAMEAS>
      <SAMEAS>
         <URI>

<VALUE>http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01</VALUE>
            <IDENTIFIER>JOL_2014_001_R_0001_01</IDENTIFIER>
            <TYPE>oj</TYPE>
         </URI>
      </SAMEAS>
      [...]
   </WORK>
   [...]
   <EXPRESSION>
      [content of expression X in given language {acc-lang}]
   <EXPRESSION>
   <MANIFESTATION>
      [content of manifestation X.01]
   <MANIFESTATION>
   <MANIFESTATION>
      [content of manifestation X.02]
   <MANIFESTATION>
   [...]
   <MANIFESTATION>
      [content of manifestation X.M]
   <MANIFESTATION>
</NOTICE>
```

3.1.3 RETRIEVE THE OBJECT-WORK NOTICE

Description

This service allows the user to search for the object notice of the given work, decoded in the given decoding language.

Only the metadata of the work are returned in the notice, with no expression or manifestation.

Request

The user must fire a GET request to the following URL:

`http://publications.europa.eu/resource/{ps-name}/{ps-id}?language={dec-lang}&filter={in_notice-only}`

where:

- `{ps-name}` is a valid production system name

- `{ps-id}` is a valid production system id identifying a work, and compatible with its `{ps-name}`
- `{dec-lang}` is a 3-chars ISO_639-3 language code identifying the decoding language to use: this is the language used for decoding the NALs associated to the notice. If decoding language is not available, the default value defined in the configuration is used.
- `{in_notice-only}` is an optional boolean that indicates if the notice contains only the properties annotated with in_notice.

  Please note: no matter what the request specifies, the response notice is always the filtered one. The filter parameter will stay for a transition period due to legacy reasons.

The following HTTP headers must be set on the request:

- `Accept:application/xml;notice=object`

Here follows some examples of valid requests that retrieve the same object, using cURL:

- `curl -H "Accept:application/xml;notice=object" "http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-01aa75ed71a1?language=eng"`
- `curl -H "Accept:application/xml;notice=object" "http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01?language=eng"`
- `curl -H "Accept:application/xml;notice=object" "http://publications.europa.eu/resource/celex/32014R0001?language=eng"`

Response

The response is an XML-formatted object notice containing the metadata of the work only.

Here follows an example of returned notice:

```
<NOTICE decoding="eng" type="object">
   <WORK>
      <URI>
         <VALUE>http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-
8e20-01aa75ed71a1</VALUE>
         <IDENTIFIER>b84f49cd-750f-11e3-8e20-01aa75ed71a1</IDENTIFIER>
         <TYPE>cellar</TYPE>
      </URI>
      <SAMEAS>
         <URI>
            <VALUE>http://publications.europa.eu/resource/celex/32014R0001</VALUE>
            <IDENTIFIER>32014R0001</IDENTIFIER>
            <TYPE>celex</TYPE>
         </URI>
      </SAMEAS>
      <SAMEAS>
         <URI>

<VALUE>http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01</VALUE>
            <IDENTIFIER>JOL_2014_001_R_0001_01</IDENTIFIER>
            <TYPE>oj</TYPE>
         </URI>
      </SAMEAS>
      [...]
   </WORK>
   [...]
</NOTICE>
```

3.1.4 RETRIEVE THE OBJECT-EXPRESSION NOTICE

Description

This service allows the user to search for the expression of the given work, decoded in the given decoding language.

The returned notice will contain the metadata of the expression in the given accept language, with no metadata of the work or manifestations.

Request

The user must fire a GET request to the following URL:

`http://publications.europa.eu/resource/{ps-name}/{ps-id}?language={dec-lang}&filter={in_notice-only}`

where:

- `{ps-name}` is a valid production system name
- `{ps-id}` is a valid production system id identifying a work, and compatible with its `{ps-name}`
- `{dec-lang}` is a 3-chars ISO_639-3 language code identifying the decoding language to use: this is the language used for decoding the NALs associated to the notice. If decoding language is not available, the default value defined in the configuration is used.
- `{in_notice-only}` is an optional boolean that indicates if the notice contains only the properties annotated with in_notice.

  Please note: no matter what the request specifies, the response notice is always the filtered one. The filter parameter will stay for a transition period due to legacy reasons.

The following HTTP headers must be set on the request:

- `Accept:application/xml;notice=object`
- `Accept-Language:{acc-lang},` where `{acc-lang}` is a 3-chars ISO_639-3 language code identifying the accept language to use: this will be used for retrieving the correct expression.

Here follows some examples of valid requests that retrieve the same object, using cURL:

- `curl -H "Accept:application/xml;notice=object" -H "Accept-Language:fra" "http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-01aa75ed71a1?language=eng"`
- `curl -H "Accept:application/xml;notice=object" -H "Accept-Language:fra" "http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01?language=eng"`
- `curl -H "Accept:application/xml;notice=object" -H "Accept-Language:fra" "http://publications.europa.eu/resource/celex/32014R0001?language=eng"`

Response

The response is an XML-formatted object notice containing the metadata of the expression only.

Here follows an example of returned notice:

```
<NOTICE decoding="eng" type="object">
   <EXPRESSION>
      <URI>
         <VALUE>http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-
8e20-01aa75ed71a1.0010</VALUE>
         <IDENTIFIER>b84f49cd-750f-11e3-8e20-01aa75ed71a1.0010</IDENTIFIER>
         <TYPE>cellar</TYPE>
      </URI>
```

```
      <SAMEAS>
         <URI>

<VALUE>http://publications.europa.eu/resource/celex/32014R0001.FRA</VALUE>
            <IDENTIFIER>32014R0001.FRA</IDENTIFIER>
            <TYPE>celex</TYPE>
         </URI>
      </SAMEAS>
      <SAMEAS>
         <URI>

<VALUE>http://publications.europa.eu/resource/uriserv/OJ.L_.2014.001.01.0001.01.FRA
</VALUE>
            <IDENTIFIER>OJ.L_.2014.001.01.0001.01.FRA</IDENTIFIER>
            <TYPE>uriserv</TYPE>
         </URI>
      </SAMEAS>
      [...]
   </EXPRESSION>
</NOTICE>
```

3.1.5 RETRIEVE THE OBJECT-MANIFESTATION NOTICE

Description

This service allows the user to search for the object notice of the given manifestation, decoded in the given decoding language.

Only the metadata of the manifestation are returned in the notice, with no work or expressions.

Request

The user must fire a GET request to the following URL:

`http://publications.europa.eu/resource/{ps-name}/{ps-id}?language={dec-lang}&filter={in_notice-only}`

where:

- `{ps-name}` is a valid production system name
- `{ps-id}` is a valid production system id identifying a manifestation, and compatible with its `{ps-name}`
- `{dec-lang}` is a 3-chars ISO_639-3 language code identifying the decoding language to use: this is the language used for decoding the NALs associated to the notice. If decoding language is not available, the default value defined in the configuration is used.
- `{in_notice-only}` is an optional boolean that indicates if the notice contains only the properties annotated with in_notice.

  Please note: no matter what the request specifies, the response notice is always the filtered one. The filter parameter will stay for a transition period due to legacy reasons.

The following HTTP headers must be set on the request:

- `Accept:application/xml;notice=object`

The following HTTP header can be set on the request:

- `Negotiate:vlist`

  If it is present, the response will include an Alternates header indicating all alternative representations of the returned object

Here follows some examples of valid requests that retrieve the same object, using cURL:

- `curl -H "Accept:application/xml;notice=object" "http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-01aa75ed71a1.0010.03?language=eng"`
- `curl -H "Accept:application/xml;notice=object" "http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01.FRA.xhtml?language=eng"`
- `curl -H "Accept:application/xml;notice=object" "http://publications.europa.eu/resource/celex/32014R0001.FRA.print?language=eng"`

Response

The response is an XML-formatted object notice containing the metadata of the manifestation only.

Here follows an example of returned notice:

```
<NOTICE decoding="eng" type="object">
   <MANIFESTATION manifestation-type="xhtml">
      <URI>
         <VALUE>http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-
8e20-01aa75ed71a1.0010.03</VALUE>
         <IDENTIFIER>b84f49cd-750f-11e3-8e20-01aa75ed71a1.0010.03</IDENTIFIER>
         <TYPE>cellar</TYPE>
      </URI>
      <SAMEAS>
         <URI>

<VALUE>http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01.FRA.xhtml</
VALUE>
            <IDENTIFIER>JOL_2014_001_R_0001_01.FRA.xhtml</IDENTIFIER>
            <TYPE>oj</TYPE>
         </URI>
      </SAMEAS>
      <SAMEAS>
         <URI>

<VALUE>http://publications.europa.eu/resource/uriserv/OJ.L_.2014.001.01.0001.01.FRA
.xhtml</VALUE>
            <IDENTIFIER>OJ.L_.2014.001.01.0001.01.FRA.xhtml</IDENTIFIER>
            <TYPE>uriserv</TYPE>
         </URI>
      </SAMEAS>
      <MANIFESTATION_TYPE type="data">
         <VALUE>xhtml</VALUE>
      </MANIFESTATION_TYPE>
      [...]
   </MANIFESTATION>
</NOTICE>
```

3.1.6 RETRIEVE THE IDENTIFIER NOTICE

3.1.6.1 As a GET request

Description

This service allows the user to retrieve the synonyms of a given resource URI.

Request

The user must fire a GET request to the following URL:

`http://publications.europa.eu/resource/{ps-name}/{ps-id}`

where:

- `{ps-name}` is a valid production system name
- `{ps-id}` is a valid production system id identifying a work, an expression, a manifestation, an item, a dossier, an event or an agent and is compatible with its `{ps-name}`

The following HTTP header must be set on the request:

- `Accept:application/xml;notice=identifiers`

Here follow some examples of valid requests that retrieve different objects, using cURL:

- curl -H "Accept:application/xml;notice=identifiers" "http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-01aa75ed71a1"
- curl -H "Accept:application/xml;notice=identifiers" http://publications.europa.eu/resource/celex/32014R0001.FRA.print
- curl -H "Accept:application/xml;notice=identifiers" http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01.FRA.fmx4.L_2014001FR.01000101.xml http://publications.europa.eu/resource/oj/JOL_2006_088_R_0063_01.fra.fmx4.L_2006088FR.01006301.xml
- curl -H "Accept:application/xml;notice=identifiers" http://publications.europa.eu/resource/pegase/11260.12796

Response

The response is an XML-formatted notice containing the URI of the cellar ID and its synonym(s).

```
<?xml version="1.0" encoding="UTF-8"?>
<NOTICE type="identifier">
   <URI>
      <VALUE>http://publications.europa.eu/resource/cellar/32a58fc1-cffa-11e1-96ce-
01aa75ed71a1.0003</VALUE>
      <TYPE>cellar</TYPE>
      <IDENTIFIER>32a58fc1-cffa-11e1-96ce-01aa75ed71a1.0003</IDENTIFIER>
   </URI>
   <SAMEAS>
      <URI>
         <VALUE>http://publications.europa.eu/resource/pegase/11260.12796</VALUE>
         <TYPE>pegase</TYPE>
         <IDENTIFIER>11260.12796</IDENTIFIER>
      </URI>
   </SAMEAS>
</NOTICE>
```

3.1.6.2 As a POST request (deprecated!)

Description

This service allows the user to retrieve the synonyms of a given list of resource URIs.

Request

The user must fire a POST request to the following URL:

`http://publications.europa.eu/webapi/getIdentifierList`

and provide the body of the request with a space separated list of valid resource URIs.

The following HTTP headers must be set on the request:

- `Accept:application/xml;notice=identifier`

Here follows some examples of valid requests using cURL:

- `curl -d http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-01aa75ed71a1`

```
      http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01 -H
      "Accept:application/xml;notice=identifier"
      "http://publications.europa.eu/webapi/getIdentifierList"
```

- `curl -d http://publications.europa.eu/resource/celex/32014R0001 -H "Accept:application/xml;notice=identifier" "http://publications.europa.eu/webapi/getIdentifierList"`

Please note that what follows the `-d` option is the body of the request, while the last argument is the service URL to call.

Response

The response is an XML-formatted notice containing the synonyms. Please note that you get 1 <OBJECT> tag per each resource URI provided on the request body, each containing the provided resource URI (<URI> tag) and its synonyms (<SAMEAS> tags).

Here follows an example of returned notice:

```
<NOTICE type="identifier">
   <OBJECT embargo-date="2014-01-04T00:00:00.000+01:00"
           in="http://publications.europa.eu/resource/celex/32014R0001">
      <URI>
         <VALUE>http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-
8e20-01aa75ed71a1</VALUE>
         <TYPE>cellar</TYPE>
         <IDENTIFIER>b84f49cd-750f-11e3-8e20-01aa75ed71a1</IDENTIFIER>
      </URI>
      <SAMEAS>
         <URI>
            <VALUE>http://publications.europa.eu/resource/celex/32014R0001</VALUE>
            <TYPE>celex</TYPE>
            <IDENTIFIER>32014R0001</IDENTIFIER>
         </URI>
      </SAMEAS>
      <SAMEAS>
         <URI>

<VALUE>http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01</VALUE>
            <TYPE>oj</TYPE>
            <IDENTIFIER>JOL_2014_001_R_0001_01</IDENTIFIER>
         </URI>
      </SAMEAS>
   </OBJECT>
   [...other <OBJECT>s, 1 per each provided resource URI on the request body...]
</NOTICE>
```

3.1.7 RETRIEVE THE RDF/XML FORMATTED METADATA FOR A GIVEN RESOURCE

Description

This service allows the user to search for the RDF (Resource Description Framework) content of the given object. The object to search for can be a work, an expression, a manifestation, a dossier, an event or an agent.

Request

The user must fire a GET request to the following URL:

`http://publications.europa.eu/resource/{ps-name}/{ps-id}?language={dec-lang}`

where:

- `{ps-name}` is a valid production system name

- `{ps-id}` is a valid production system id identifying a work, and compatible with its `{ps-name}`

The following HTTP headers may be set on the request:

- `Accept:application/rdf+xml`

  In this case, the resulting RDF notice will contain the direct and inferred triples.

- `Accept:application/rdf+xml;notice=non-inferred`

  In this case, the inferred triples will be excluded from the resulting RDF notice.

- `Negotiate:vlist`

  If it is present, the response will include an Alternates header indicating all alternative representations of the returned object. Currently, this header is supported only for requests on manifestation level.

If the `Accept` header is not present, * or */* and the production identifier matches a WEM object, it will behave like if set to `Accept:application/rdf+xml`.

Here follows an example of valid request that retrieve the same RDF, using cURL:

- `curl "http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-01aa75ed71a1"`
- `curl "http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01"`
- `curl "http://publications.europa.eu/resource/celex/32014R0001"`
- `curl -H "Accept: application/rdf+xml" "http://publications.europa.eu/resource/celex/32014R0001"`
- `curl -H "Accept:" "http://publications.europa.eu/resource/celex/32014R0001"`
- `curl -H "Accept: *" "http://publications.europa.eu/resource/celex/32014R0001"`
- `curl -H "Accept:*/*" "http://publications.europa.eu/resource/celex/32014R0001"`

Response

The response is an XML-formatted sheet containing the RDF metadata of the object.

Here follows an example of returned notice:

```
<rdf:RDF [...] >
   <rdf:Description
rdf:about="http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01.ELL">
    <rdf:type
rdf:resource="http://publications.europa.eu/ontology/cdm#expression"/>
   </rdf:Description>
   <rdf:Description
rdf:about="http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-
01aa75ed71a1.0022">
    <owl:sameAs
rdf:resource="http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01.SLV"
/>
   </rdf:Description>
   <rdf:Description
rdf:about="http://publications.europa.eu/resource/celex/32013R1421">
      <j.0:resource_legal_consolidated_by_act_consolidated
rdf:resource="http://publications.europa.eu/resource/celex/02012R0978-20141001"/>
    <j.0:consolidated_by
rdf:resource="http://publications.europa.eu/resource/celex/02012R0978-20141001"/>
```

```
    <j.0:resource_legal_consolidated_by_act_consolidated
rdf:resource="http://publications.europa.eu/resource/celex/02012R0978-20150101"/>
    <j.0:consolidated_by
rdf:resource="http://publications.europa.eu/resource/celex/02012R0978-20150101"/>
   </rdf:Description>
   [...]
</rdf:RDF>
```

3.1.8 RETRIEVE THE RDF/XML FORMATTED METADATA OF THE TREE WHOSE ROOT IS A GIVEN RESOURCE

Description

This service allows the user to search for the RDF (Resource Description Framework) tree whose root is the given object. The object to search for can be a work, a dossier or an agent.

Request

The user must fire a GET request to the following URL:

`http://publications.europa.eu/resource/{ps-name}/{ps-id}?language={dec-lang}`

where:

- `{ps-name}` is a valid production system name
- `{ps-id}` is a valid production system id identifying a work, and compatible with its `{ps-name}`

The following HTTP headers may be set on the request:

- `Accept:application/rdf+xml;notice=tree`

  In this case, the resulting RDF notice will contain the direct and inferred triples.

- `Accept:application/rdf+xml;notice=non-inferred-tree`

  In this case, the inferred triples will be excluded from the resulting RDF notice.

Here follows some examples of valid requests that retrieve the same RDF, using cURL :

- `curl -H "Accept:application/rdf+xml;notice=tree" "http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-01aa75ed71a1"`
- `curl -H "Accept:application/rdf+xml;notice=tree" "http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01"`
- `curl -H "Accept:application/rdf+xml;notice=tree" "http://publications.europa.eu/resource/celex/32014R0001"`

Response

The response is an XML-formatted sheet containing the RDF metadata of the tree.

Here follows an example of returned notice:

```
<rdf:RDF [...] >
   <rdf:Description
rdf:about="http://publications.europa.eu/resource/authority/language/EST">
    <rdf:type rdf:resource="http://publications.europa.eu/ontology/cdm#language"/>
    <rdf:type rdf:resource="http://www.w3.org/2004/02/skos/core#Concept"/>
    <j.1:inScheme
rdf:resource="http://publications.europa.eu/resource/authority/language"/>
    <j.0:language_used_by_expression
rdf:resource="http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01.EST"
/>
  </rdf:Description>
```

```
  <rdf:Description
rdf:about="http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-
01aa75ed71a1.0005.01">
    <j.2:metsStructSuperDiv
rdf:resource="http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-
8e20-01aa75ed71a1.0005"/>
    <j.2:lastModificationDate
rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">2014-01-
04T08:10:26.028+01:00</j.2:lastModificationDate>
    <owl:sameAs
rdf:resource="http://publications.europa.eu/resource/uriserv/OJ.L_.2014.001.01.0001
.01.ELL.pdfa1a"/>
    <j.0:manifestation_has_item
rdf:resource="http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-
8e20-01aa75ed71a1.0005.01/DOC_1"/>
    [...]
  </rdf:Description>
   [...]
</rdf:RDF>
```

3.1.9 RETRIEVE CONTENT STREAMS

Description

This service allows the user to retrieve the content stream of the manifestation belonging to the given work and to the expression in the given accept language, and which contains at least 1 content stream of the given accept format.

Request

The user must fire a GET request to the following URL:

`http://publications.europa.eu/resource/{ps-name}/{ps-id}?language={dec-lang}`

where:

- `{ps-name}` is a valid production system name
- `{ps-id}` is a valid production system id identifying a work, and compatible with its `{ps-name}`

The following HTTP headers must be set on the request:

- `Accept:{mime-type},` where `{mime-type}` is a valid (or a comma-separated list of) mimetype that identify the format of the content stream to return. Possible values are:
  - `application/epub+zip`
  - `application/msword`
  - `application/pdf,application/pdf;type=pdf1x`
  - `application/pdf;type=pdfa1a`
  - `application/pdf;type=pdfa1b`
  - `application/pdf;type=pdfx`
  - `application/rdf+xml`
  - `application/sparql-query`
  - `application/sparql-results+xml`
  - `application/vnd.amazon.ebook`
  - `application/vnd.ms-excel`
  - `application/vnd.ms-powerpoint`

- application/vnd.openxmlformats-officedocument.presentationml.presentation
- application/vnd.openxmlformats-officedocument.presentationml.slideshow
- application/vnd.openxmlformats-officedocument.spreadsheetml.sheet
- application/vnd.openxmlformats-officedocument.wordprocessingml.document.main+xml
- application/x-mobipocket-ebook
- application/xhtml+xml
- application/xhtml+xml;type=simplified
- application/xml
- application/xml;type=fmx2,text/sgml;type=fmx2
- application/xml;type=fmx3,text/sgml;type=fmx3
- application/xml;type=fmx4
- application/xslt+xml
- application/zip
- image/gif
- image/jpeg
- image/png
- image/tiff,image/tiff-fx
- text/html
- text/html;type=simplified
- text/plain
- text/rtf
- text/sgml

- `Accept-Language:{acc-lang}`, where `{acc-lang}` is a 3-chars ISO_639-3 language code identifying the accept language to use: this will be used for retrieving the correct expression
- `Accept-Max-Cs-Size:{size}`, where `{size}` is a positive integer (max. value = 2^{63}-1) which specifies the max. content stream size in bytes. If the actual content stream size is bigger than specified, a “406 - Not Acceptable” response is given.

Here follows some examples of valid request that retrieve the same content stream, using cURL:

- `curl -H "Accept:application/xhtml+xml;" -H "Accept-Language:fra" "http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-01aa75ed71a1"`
- `curl -H "Accept:application/xhtml+xml;" -H "Accept-Language:fra" "http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01"`
- `curl -H "Accept:application/xhtml+xml;" -H "Accept-Language:fra" –H "Accept-Max-Cs-Size:209715200" "http://publications.europa.eu/resource/celex/32014R0001"`

Response

The associated content stream.

3.1.10 RETRIEVE CONTENT STREAM COLLECTIONS

Description

This service allows the user to retrieve a collection (in zip or list format) of the content streams of the manifestation belonging to the given work and to the expression in the given accept language, and which contains at least 1 content stream of the given accept format.

Request

The user must fire a GET request to the following URL:

`http://publications.europa.eu/resource/{ps-name}/{ps-id}?language={dec-lang}`

where:

- `{ps-name}` is a valid production system name
- `{ps-id}` is a valid production system id identifying a work, and compatible with its `{ps-name}`

The following HTTP headers must be set on the request:

- `Accept:{mime-type},` where `{mime-type}` is a valid (or a comma-separated list of) mimetype that identify the format of the content stream to return. Possible values are:
  - `application/list;mtype={manifestation-type}`
  - `application/zip;mtype={manifestation-type}`

  The mtype token carries the `{manifestation-type}`, which must be set to the value of cdm:manifestation_type of the desired manifestation
- `Accept-Language:{acc-lang},` where `{acc-lang}` is a 3-chars ISO_639-3 language code identifying the accept language to use: this will be used for retrieving the correct expression

Here follows some examples of valid request that retrieve the same content stream, using cURL:

- ```
  curl -H "Accept:application/zip;mtype=fmx4" -H "Accept-Language:fra"
  http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-
  01aa75ed71a1
  ```
- ```
  curl -H "Accept:application/list;mtype=fmx4" -H "Accept-Language:fra"
  "http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-
  01aa75ed71a1"
  ```

Response

The associated content streams in the requested format:

- zip: a zip file containing all content stream files of the requested manifestation
- list: an html list containing all content stream file names of the requested manifestation

Note: If the given resource is a manifestation and the mtype token does not match its type, the mtype token is ignored and content streams of the given manifestation are returned.

3.2 NAL/EUROVOC SERVICES

We describe hereby the available services for retrieving the information related to the NAL/EUROVOC objects.

Some of the services below rely heavily on the notions of:

- **concept**, which is the class defined by the resource URI http://publications.europa.eu/ontology/cdm#concept.

  It is the superclass of all concepts used in Cellar's ontology and a direct subclass of the SKOS concept (`http://www.w3.org/2004/02/skos/core#Concept`), thus it can be seen as the topmost class of Cellar's ontology

- **concept scheme**, which has the same meaning as the SKOS concept scheme (`http://www.w3.org/2004/02/skos/core#ConceptScheme`): an aggregation of one or more concepts.

  Semantic relationships (links) between those concepts may also be viewed as part of a concept scheme. This definition is, however, meant to be suggestive rather than restrictive, and there is some flexibility in the formal data model of the Cellar.

3.2.1 RETRIEVE A DUMP

Description

This service allows the user to retrieve the complete dump of a NAL or EUROVOC object.

Request

The user must fire a GET request to the following URL:

`http://publications.europa.eu/webapi/authority-table?object={object-id}&DS={ds}`

where:

- `{object-id} can be:`
  - either the last segment of a NAL's resource URI. For example, if the NAL resource URI is `http://publications.europa.eu/resource/authority/fd_010`, the correct value for `{object-id}` is `fd_010`
  - either `EUROVOC`
- `{ds}` identifies the format of the content stream containing the desired dump. It can assume the following values:
  - `SKOS:` in this case, the complete SKOS/RDF dump will be returned. The response content type will be `application/rdf+xml`
  - `XML:` in this case, the dump will be embedded into a ZIP package, containing all the XML files of the NAL/EUROVOC object. The response content type will be `application/zip`.

Here follows some examples of valid requests:

- http://publications.europa.eu/webapi/authority-table?object=fd_010&DS=XML
- http://publications.europa.eu/webapi/authority-table?object=fd_010&DS=SKOS
- http://publications.europa.eu/webapi/authority-table?object=EUROVOC&DS=XML

Response

As already mentioned, we have 2 different types of response:

- if the user specified on the request `DS=SKOS`, the response content will be of type `application/rdf+xml`, and the dump will be returned as an XML-formatted SKOS/RDF sheet

- if the user specified on the request `DS=XML`, the response content will be of type `application/zip`, and the dump will be returned as a ZIP package containing all the XML files of the NAL/EUROVOC object.

3.2.2 RETRIEVE THE SUPPORTED LANGUAGES

Description

This service allows the user to retrieve the supported languages of the system. Also, the user may ask for the supported languages of a particular NAL/EUROVOC concept scheme.

Request

The user must fire a GET request to the following URL:

`http://publications.europa.eu/webapi/{type}/getSupportedLanguages?concept_scheme={cs-uri}`

where:

- `{type}` can be either `nal` or `eurovoc`, depending on whether the user wants to retrieve the supported languages for NAL or EUROVOC objects, respectively
- `{cs-uri}` is the resource URI of the NAL/EUROVOC concept scheme.

  This parameter is not mandatory: if not specified, all supported languages of the system will be retrieved.

Here follows some examples of valid requests:

- `http://publications.europa.eu/webapi/nal/getSupportedLanguages`
- `http://publications.europa.eu/webapi/nal/getSupportedLanguages?concept_scheme=http%3A%2F%2Fpublications.europa.eu%2Fresource%2Fauthority%2Ffd_330"`
- `http://publications.europa.eu/webapi/eurovoc/getSupportedLanguages`

Response

The list of supported languages in JSON format. For more information about JSON format, please see Annexe 3.

Example:

```
[
  {
    "code": "mlt"
  },
  {
    "code": "deu"
  },
  [...other languages]
]
```

3.2.3 RETRIEVE A CONCEPT SCHEME

Description

This service allows the user to retrieve a NAL or EUROVOC concept scheme.

Request

The user must fire a GET request to the following URL:

http://publications.europa.eu/webapi/{type}/getConceptScheme?concept_Scheme={cs-uri}

where

- `{type}` can be either `nal` or `eurovoc`, depending on whether the user wants to retrieve a NAL or an EUROVOC concept scheme, respectively
- `{cs-uri}` is the resource URI of the NAL/EUROVOC concept scheme.

  This parameter is mandatory only for NALs (that is, when `{type}` is `nal`): if not specified for EUROVOCs (`{type}` is `eurovoc`), it defaults to `http://eurovoc.europa.eu/100141`.

Here follows some examples of valid requests:

- http://publications.europa.eu/webapi/nal/getConceptScheme?concept_scheme=http%3A%2F%2Fpublications.europa.eu%2Fresource%2Fauthority%2Ffd_330
- http://publications.europa.eu/webapi/eurovoc/getConceptScheme?concept_scheme=http%3A%2F%2Feurovoc.europa.eu%2F100225
- http://publications.europa.eu/webapi/eurovoc/getConceptScheme

Response

The concept scheme in JSON format.

Example:

```
{
  "date": null,
  "lastModified": null,
  "version": null,
  "uri": {
    "uri": "http://eurovoc.europa.eu/100225"
  },
  "labels": [
    {
      "language": "ron",
      "string": "3611 ştiinţe umaniste"
    },
    {
      "language": "hun",
      "string": "3611 humán tudományok"
    },
    [...other labels]
  ]
}
```

3.2.4 RETRIEVE THE CONCEPT SCHEMES

Description

This service allows the user to retrieve all the concept schemes of NALs or EUROVOCs.

Request

The user must fire a GET request to the following URL:

http://publications.europa.eu/webapi/{type}/getConceptSchemes

where `{type}` can be either `nal` or `eurovoc`, depending on whether the user wants to retrieve the concept schemes of NALs or EUROVOCs, respectively.

Here follows some examples of valid requests:

- http://publications.europa.eu/webapi/nal/getConceptSchemes
- http://publications.europa.eu/webapi/eurovoc/getConceptSchemes

Response

The list of concept schemes in JSON format.

Example:


```
[
  {
    "date": null,
    "lastModified": null,
    "version": null,
    "uri": {
      "uri": "http://eurovoc.europa.eu/100225"
    },
    "labels": [
      {
        "language": "ron",
        "string": "3611 ştiinţe umaniste"
      },
      {
        "language": "hun",
        "string": "3611 humán tudományok"
      },
      [...other labels]
    ]
  },
  {
    "date": null,
    "lastModified": null,
    "version": null,
    "uri": {
      "uri": "http://eurovoc.europa.eu/100226"
    },
    "labels": [
      {
        "language": "ron",
```

```
        "string": "4006 organizarea afacerilor"
      },
      {
        "language": "hun",
        "string": "4006 gazdasági szervezetek"
      },
      [...other labels]
    ]
  },
  [...other concepts schemes]
]
```

3.2.5 RETRIEVE A CONCEPT

Description

This service allows the user to retrieve the translation of a given concept into a specified language.

Request

The user must fire a GET request to the following URL:

`http://publications.europa.eu/webapi/{type}/getConcept?concept_uri={con-uri}&language={lang}`

where:

- `{type}` can be either `nal` or `eurovoc`, depending on whether the user wants to retrieve the translation of a NAL or EUROVOC concept, respectively
- `{con-uri}` is the resource URI of the NAL/EUROVOC concept
- `{lang}` is a 3-chars ISO_639-3 language code identifying the language the user wants to translate the concept with.

Here follows some examples of valid requests:

- `http://publications.europa.eu/webapi/nal/getConcept?concept_uri=http%3A%2F%2Fpublications.europa.eu%2Fresource%2Fauthority%2Ffd_330&language=eng`
- `http://publications.europa.eu/webapi/eurovoc/getConcept?concept_uri=http%3A%2F%2Feurovoc.europa.eu%2F100225&language=fra`

Response

The translated concept in JSON format.

Example:

```
[
  {
    "language": "fra",
    "identifier": "3928",
    "notations": [

    ],
```

```
    "uri": {
      "uri": "http://eurovoc.europa.eu/3928"
    },
    "prefLabel": {
      "language": "fra",
      "string": "sciences du comportement"
    },
    "altLabels": [
      "psychologie du comportement",
      "behaviorisme"
    ],
    "hiddenLabels": [
      "comportement, psychologie du",
      "comportement, sciences du"
    ]
  }
]
```


3.2.6 RETRIEVE THE CONCEPT RELATIVES

Description

This service allows the user to retrieve the list of concepts having a specific semantic relation with the given concept.

Request

The user must fire a GET request to the following URL:

`http://publications.europa.eu/webapi/{type}/getConceptRelatives?concept_uri={con-uri}&relation_uri={rel-uri}&language={lang}`

where:

- `{type}` can be either `nal` or `eurovoc`, depending on whether the user wants to retrieve the concept relatives of a NAL or EUROVOC concept, respectively
- `{con-uri}` is the resource URI of the NAL/EUROVOC concept of which retrieving the concept relatives
- `{rel-uri}` is the resource URI of the SKOS relation scheme to use, namely:
  - `http://www.w3.org/2004/02/skos/core#broader`: to use in order to retrieve the concepts that are more general in meaning than the given concept. Broader concepts are typically rendered as parents in a concept hierarchy
  - `http://www.w3.org/2004/02/skos/core#narrower`: to use in order to retrieve the concepts that are more specific in meaning than the given concept. Narrower concepts are typically rendered as children in a concept hierarchy
  - `http://www.w3.org/2004/02/skos/core#related`: to use in order to retrieve the concepts that have an associative semantic relationship with the given concept
- `{lang}` is a 3-chars ISO_639-3 language code identifying the language the user wants to retrieve the concept relatives with.

Here follows some examples of valid requests:

- http://publications.europa.eu/webapi/nal/getConceptRelatives?concept_uri=http%3A%2F%2Fpublications.europa.eu%2Fresource%2Fauthority%2Ffd_330&relation_uri=http%3A%2F%2Fwww.w3.org%2F2004%2F02%2Fskos%2Fcore%23broader&language=eng
- http://publications.europa.eu/webapi/eurovoc/getConceptRelatives?concept_uri=http%3A%2F%2Feurovoc.europa.eu%2F100225&relation_uri=http%3A%2F%2Fwww.w3.org%2F2004%2F02%2Fskos%2Fcore%23narrower&language=fra

Response

The list of concepts that have a semantic relation with the given concept, in JSON format.

Example:


```
[
  {
    "language": "fra",
    "identifier": "3928",
    "notations": [

    ],
    "uri": {
      "uri": "http://eurovoc.europa.eu/3928"
    },
    "prefLabel": {
      "language": "fra",
      "string": "sciences du comportement"
    },
    "altLabels": [
      "psychologie du comportement",
      "behaviorisme"
    ],
    "hiddenLabels": [
      "comportement, psychologie du",
      "comportement, sciences du"
    ]
  },
  {
    "language": "fra",
    "identifier": "3956",
    "notations": [

    ],
    "uri": {
      "uri": "http://eurovoc.europa.eu/3956"
```

```
    },
    "prefLabel": {
      "language": "fra",
      "string": "sciences sociales"
    },
    "altLabels": [
      "sciences humaines"
    ],
    "hiddenLabels": [
      "sociales, sciences",
      "humaines, sciences"
    ]
  },
  [...other concepts]
]
```

3.2.7 RETRIEVE THE TOP CONCEPTS

Description

This service allows the user to retrieve the top concepts of a given concept scheme in a specified language.

A top concept is a concept that is topmost in the broader/narrower concept hierarchies for a given concept scheme, providing an entry point to these hierarchies.

Request

The user must fire a GET request to the following URL:

`http://publications.europa.eu/webapi/{type}/getTopConcepts?concept_scheme={cs-uri}&language={lang}`

where:

- `{type}` can be either `nal` or `eurovoc`, depending on whether the user wants to retrieve the top concepts of a NAL or EUROVOC concept, respectively
- `{cs-uri}` is the resource URI of the NAL/EUROVOC concept scheme of which retrieving the top concepts
- `{lang}` is a 3-chars ISO_639-3 language code identifying the language the user wants to retrieve the top concepts with.

Here follows some examples of valid requests:

- `http://publications.europa.eu/webapi/nal/getTopConcepts?concept_scheme=http%3A%2F%2Fpublications.europa.eu%2Fresource%2Fauthority%2Ffd_330&language=eng`
- `http://publications.europa.eu/webapi/eurovoc/getTopConcepts?concept_scheme=http%3A%2F%2Feurovoc.europa.eu%2F100225&language=fra`

Response

The list of top concepts in JSON format.

Example:

```
[
```

```
{
  "language": "fra",
  "identifier": "3928",
  "notations": [

  ],
  "uri": {
    "uri": "http://eurovoc.europa.eu/3928"
  },
  "prefLabel": {
    "language": "fra",
    "string": "sciences du comportement"
  },
  "altLabels": [
    "psychologie du comportement",
    "behaviorisme"
  ],
  "hiddenLabels": [
    "comportement, psychologie du",
    "comportement, sciences du"
  ]
},
{
  "language": "fra",
  "identifier": "3956",
  "notations": [

  ],
  "uri": {
    "uri": "http://eurovoc.europa.eu/3956"
  },
  "prefLabel": {
    "language": "fra",
    "string": "sciences sociales"
  },
  "altLabels": [
    "sciences humaines"
  ],
  "hiddenLabels": [
    "sociales, sciences",
```

```
      "humaines, sciences"
    ]
  },
  [...other concepts]
]
```

3.2.8 RETRIEVE THE DOMAINS

Description

This service allows the user to retrieve the domains facets of the EUROVOC thesaurus.

Request

The user must fire a GET request to the following URL:

http://publications.europa.eu/webapi/eurovoc/getDomains

Response

The list of domains in JSON format.

Example:

```
[
  {
    "identifier": "28",
    "uri": {
      "uri": "http://eurovoc.europa.eu/100149"
    },
    "conceptSchemes": [
      {
        "uri": "http://eurovoc.europa.eu/100212"
      },
      {
        "uri": "http://eurovoc.europa.eu/100213"
      },
      [...other concept scheme URIs]
    ],
    "labels": [
      {
        "language": "ron",
        "string": "28 PROBLEME SOCIALE"
      },
      {
        "language": "hun",
        "string": "28 TÁRSADALMI KÉRDÉSEK"
      },
```

```
      [...other labels]
    ]
  },
  {
    "identifier": "24",
    "uri": {
      "uri": "http://eurovoc.europa.eu/100148"
    },
    "conceptSchemes": [
      {
        "uri": "http://eurovoc.europa.eu/100200"
      },
      {
        "uri": "http://eurovoc.europa.eu/100203"
      },
      [...other concept scheme URIs]
    ],
    "labels": [
      {
        "language": "hr",
        "string": "24 FINANCIJE"
      },
      {
        "language": "ron",
        "string": "24 FINANTE"
      },
      [...other labels]
    ]
  },
  [...other domains]
]
```


3.3 NOTIFICATIONS: RSS AND ATOM FEEDS

This notification service provides information about the ingesting of documents, the loading of NALs and the loading of ontologies in the form of an RSS or Atom feed. By accessing these feeds, it is possible to get a complete history of the performed actions.

Please note: as this feature is introduced in Cellar 6.2.0, it is yet not possible to retrieve information of actions executed before the installation of this Cellar release.

3.3.1 REQUEST

To specify if the response is given in RSS or in Atom format, the HTTP header `Accept:{accept-type}` must be specified, where `{accept-type}` is a string which may assume the following values:

- `application/rss+xml`, in which case the Cellar will provide the results as an RSS feed
- `application/atom+xml`, in which case the Cellar will provide the results as an ATOM feed

If this header is not set, it defaults to `application/rss+xml`.

To request a feed, the following URL must be used:

`http://[CELLAR_IP]:[CELLAR_PORT]/[CELLAR_CONTEXT]/webapi/notification/{type}?{parameters}`

`type` must be set to one of the three available notification types: `ingestion`, `nal` or `ontology`. The next chapters describe the different notification types and their possible `parameters`.

Date parameters must match one of these 3 ISO8601 standard formats: yyyy-MM-dd (*2013-12-02*), yyyy-MM-dd'T'HH:mm:ss (*2013-12-02T09:24:22*), yyyy-MM-dd'T'HH:mm:ssZZ (*2013-12-02T09:24:22-01:00*).

Note: only the `startDate` parameter is mandatory; if an optional parameter is not set, no filter is applied.

3.3.1.1 Ingestion notification

Provides an overview of all ingestion actions filtered by the given parameters.

- `startDate`: Defines the date (inclusive) since which the ingestion notifications shall be retrieved
- `endDate`: Defines the date (inclusive) until which the ingestion notifications shall be retrieved
- `type`: A string value, either `CREATE`, `UPDATE` or `DELETE`. It defines the type of ingestion to be retrieved.
- `wemiClasses`: A comma-separated list of WEMI classes: `work`, `expression`, `manifestation`, `item`, `dossier`, `event` or `agent`.
- `page`: The number of the page on the feed to display, should the total number of entries returned be higher than 1000 (defined in property `cellar.service.notification.itemsPerPage`). This parameter may be used to page large results by firing subsequent requests and setting incremental values on this parameter. If not set, page 1 is returned.

3.3.1.2 NAL notification

Provides an overview of all NAL loading actions filtered by the given parameters.

- `startDate`, `endDate`, `page`: same as explained above.

3.3.1.3 Ontology notfication

Provides an overview of all NAL loading actions filtered by the given parameters.

- `startDate`, `endDate`, `page`: same as explained above.

3.3.1.4 Example requests:

Retrieve: the 3rd page of of the RSS feed containing the updates of works and events occurred from the 1st January of 2012 until the 31st December 2012:

- `curl -H "Accept:application/rss+xml" http://publications.europa.eu/webapi/notification/ingestion?startDate=2012-01-01&endDate=2012-12-31&type=UPDATE&wemiClasses=work,event&page=3`

Retrieve the 1st page of the ATOM feed containing the creations and updates of all types of entities occurred from the 31st December 2012 until now:

- `curl -H "Accept:application/atom+xml" http://publications.europa.eu/webapi/notification/ingestion?startDate=2012-12-31`

Retrieve the 3rd page of the RSS feed containing the successful NAL updates occurred from the 1st January of 2012 until the 31st December 2012:

- `curl -H "Accept:application/rss+xml" http://publications.europa.eu/webapi/notification/nal?startDate=2012-01-01&endDate=2012-12-31page=3`

Retrieve the 1st page of the ATOM feed containing the successful ontology updates occurred from the 31st December 2012 until now:

- `curl -H "Accept:application/atom+xml" http://publications.europa.eu/webapi/notification/ontology?startDate=2012-12-31`

3.3.2 RESPONSE

The response contains the following information (no matter what format).

- *title*: Title of the feed, each feed (ingestion, NAL, ontology) has its own title.
- *startDate*: Same as `startDate` parameter
- *endDate*: Same as `endDate` parameter; current date if `endDate` was not set or defined in the future.
- *type* (only ingestion): Same as `type` parameter in ingestion feed request.
- *page*: Cardinal number of the current page of the results.
- *moreEntries*: If true, the result has been paged and more entries that satisfy the request have been found. Subsequent requests should be fired with increasing page numbers.

The items (RSS) / entries (Atom) of the feed answer differ for each notification type. They are described below.

3.3.2.1 Ingestion items/entries

- *guid*: The cellar ID of the ingested element.
- *type*: The ingestion type of the ingested element.
- *classes*: The class hierarchy of the ingested element, the top class ist the most specific, the bottom one the most general.
- *identifiers*: The sameases of the ingested element.
- *date*: The ingestion date and time.

3.3.2.2 NAL items/entries,

- *guid*: The URI of the loaded NAL.
- *version*: The version (creation date) of the NAL.
- *date*: The date and time of the NAL loading.

3.3.2.3 Ontology items/entries

- *guid*: The URI of the loaded ontology.
- *version*: The version of the loaded ontology.
- *date*: The date and time of the ontology loading.

3.3.2.4 Example response (RSS ingestion)

```
<rss version="2.0"
  xmlns:notifReq="http://publications.europa.eu/rss/notificationRequest">
  xmlns:notifEntry="http://publications.europa.eu/rss/notificationEntry">
  <channel>
    <title>Ingestion Notification Messages Response</title>
    <notifReq:startDate>2012-01-01T00:00:00+01:00</notifReq:startDate>
    <notifReq:endDate>2012-12-31T00:00:00+01:00</notifReq:endDate>
    <notifReq:type>UPDATE</notifReq:type>
    <notifReq:page>1</notifReq:page>
    <notifReq:moreEntries>false</notifReq:moreEntries>
    <item>
      <guid isPermaLink="false">cellar:ca753ae9-cf80-11e2-859e-01aa75ed71a1</guid>
      <notifEntry:type>UPDATE</notifEntry:type>
      <notifEntry:classes>
        <notifEntry:class>http://publications.europa.eu/ontology/cdm#case-law_national</notifEntry:class>
        <notifEntry:class>http://publications.europa.eu/ontology/cdm#case-law</notifEntry:class>
        <notifEntry:class>http://publications.europa.eu/ontology/cdm#resource_legal</notifEntry:class>
        <notifEntry:class>http://publications.europa.eu/ontology/cdm#work</notifEntry:class>
      </notifEntry:classes>
      <notifEntry:identifiers>
        <notifEntry:identifier>oj:JOL_2012_154_R_0012_01</notifEntry:identifier>
        <notifEntry:identifier>celex:32006D0241</notifEntry:identifier>
      </notifEntry:identifiers>
      <notifEntry:date>2012-06-11T09:13:58+01:00</notifEntry:date>
    </item>
  </channel>
</rss>
```

4 ANNEXES

4.1 ANNEX 1: LIST OF ISO_639-3 CODES OF SUPPORTED EUROPEAN LANGUAGES

The Cellar supports the European languages identified by the following ISO_639-3 codes:

ISO_639-3 code	Language
bul	Bulgarian
ces	Czech
dan	Danish
deu	German
ell	Modern Greek
eng	English
est	Estonian
fin	Finnish
fra	French
gle	Irish
hrv	Croatian
hun	Hungarian
isl	Icelandic
ita	Italian
lav	Latvian
lit	Lithuanian
mlt	Maltese
nld	Dutch
nor	Norwegian
pol	Polish
por	Portuguese
ron	Romanian, Moldavian, Moldovan
slk	Slovak
slv	Slovene
spa	Spanish, Castillian
swe	Swedish

Table 5 – Supported European languages with their ISO_639-3 codes

4.2 ANNEX 2: CURL

cURL (*Client URL Request Library*) is a computer software providing command-line tool for transferring data using various protocols, the most important of which, for our purposes, is HTTP/HTTPS.

The present document uses cURL for depicting all the examples of HTTP requests: cURL is preferable to in-browser or other graphical tools, as:

1) it is independent from the OS
2) the way a browser allows the user to build the HTTP requests may differ from browser to browser

3) its syntax does not depend on the version used, while the browser may change during time the way it represents the HTTP request

4) its syntax is simple and direct to the goal.

Basic use of cURL involves simply typing curl at the command line, followed by the URL of the output to retrieve. For example, to retrieve the example.com homepage, type:

```
curl "http://www.example.com"
```

For specifying an HTTP request header it is enough to type:

```
curl –H "myHeaderName:myHeaderValue" "http://www.example.com"
```

where `myHeaderName` is the name of the header and `myHeaderValue` is its value.

This is enough for our purposes: for more information, please refer to cURL home page at http://curl.haxx.se/.

4.3 ANNEX 3: JSON

JSON (*JavaScript Object Notation*) is a lightweight data-interchange format.

It has several advantages:

1) it is easy for humans to read and write

2) it is easy for machines to parse and generate

3) it is based on a subset of the JavaScript Programming Language, used worldwide

4) it is a text format that is completely language independent, but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others.

These properties make JSON an ideal data-interchange language.

JSON's basic types are:

- *Number* (double precision floating-point format in JavaScript, generally depends on implementation)
- *String* (double-quoted Unicode, with backslash escaping)
- *Boolean* (true or false)
- *Array* (an ordered sequence of values, comma-separated and enclosed in square brackets; the values do not need to be of the same type)
- *Object* (an unordered collection of key:value pairs with the ':' character separating the key and the value, comma-separated and enclosed in curly braces; the keys must be strings and should be distinct from each other)
- *null* (empty)

Non-significant white space may be added freely around the "structural characters" (i.e. the brackets "[{]}", colon ":" and comma ",").

The following example shows the JSON representation of an object that describes a person. The object has string fields for first name and last name, a number field for age, contains an object representing the person's address, and contains a list (an array) of phone number objects.

```
{
    "firstName": "John",
    "lastName": "Smith",
```

```
    "age": 25,
    "address": {
        "streetAddress": "21 2nd Street",
        "city": "New York",
        "state": "NY",
        "postalCode": "10021"
    },
    "phoneNumber": [
        {
            "type": "home",
            "number": "212 555-1234"
        },
        {
            "type": "fax",
            "number": "646 555-4567"
        }
    ]
}
```

4.4 ANNEX 4: OWL

The Common Data Model is expressed formally as an *ontology* – a set of concepts within a domain, and the relationships among those concepts – according to a format called the *Web Ontology Language (OWL)*. The ontology formally defines the various classes and properties and assigns unique URIs to them that reside under the URI:

`http://publications.europa.eu/ontology/cdm`

The ontology also defines certain *inferred* behaviours for classes and properties. For example, being a member of a subclass, e.g. a directive, implies being a member also of its superclasses, e.g. secondary legislation and resource legal. Also, if act A repeals another act B it is possible to infer that B is repealed by A. Inferred classes and properties are also exposed by the Cellar alongside explicitly provided ones.

The last version of Cellar CDM is accessible via the WIKI:

`http://www.cc.cec/wikis/display/OP/CMR+Common+Data+Model`