

EDI Client

Fabio Pavesi

June 21, 2023

# Contents

<b>1</b>	<b>Introduction</b>	<b>4</b>
<b>2</b>	<b>Templates</b>	<b>4</b>
2.1	Settings . . . . .	4
2.1.1	userInterfaceLanguage . . . . .	4
2.1.2	metadataLanguage . . . . .	4
2.1.3	metadataEndpoint . . . . .	4
2.1.4	sparqlEndpoint . . . . .	4
2.1.5	requiresValidation . . . . .	5
2.1.6	baseDocument . . . . .	5
2.1.7	xsltChain . . . . .	5
2.2	endpointTypes . . . . .	5
2.3	datasources . . . . .	5
2.3.1	sparql . . . . .	5
2.3.2	codelist . . . . .	6
2.3.3	singleton . . . . .	6
2.4	datasources . . . . .	6
2.4.1	sparql . . . . .	6
2.4.2	codelist . . . . .	7
2.4.3	singleton . . . . .	7
<b>3</b>	<b>Data Types</b>	<b>7</b>
3.1	Base data types . . . . .	7
3.1.1	text . . . . .	7
3.1.2	string . . . . .	7
3.1.3	URN . . . . .	8
3.1.4	URI . . . . .	8
3.1.5	URL . . . . .	8
3.1.6	int . . . . .	8
3.1.7	float . . . . .	8
3.1.8	real . . . . .	9
3.1.9	double . . . . .	9
3.1.10	codelist . . . . .	9
3.1.11	autoCompletion . . . . .	10
3.1.12	boolean . . . . .	10
3.2	Special case data types . . . . .	10
3.2.1	label . . . . .	10
3.2.2	image . . . . .	10
3.2.3	qrcode . . . . .	10
3.2.4	select . . . . .	11
3.2.5	copy . . . . .	11
3.2.6	function . . . . .	11
3.2.7	ref . . . . .	11

3.2.8	autonumber . . . . .	12
3.2.9	hidden . . . . .	12
3.2.10	date . . . . .	12
3.2.11	dateRange . . . . .	12
3.2.12	boundingBox . . . . .	13

# 1 Introduction

EDI is a template-based metadata editor.

## 2 Templates

Templates define the rules for the standards the metadatum it represents must comply to.

From a standards perspective it specifies all destination XPaths for the elements and items it contains and whether said destinations can be repeated or not.

Templates also specify how the metadata editor will behave, both in terms of validation and of manner elements and items are displayed.

Every template must contain x sections:

- settings
- endpointTypes
- datasources
- group

Templates are written in XML.

An XSD schema [is available for validating at this URL](#)

A collection of sample templates is available here . [\\_\\_\\_\\_\\_](#)

### 2.1 Settings

#### 2.1.1 userInterfaceLanguage

Labels can be defined in as many languages as required, by using the *xml:lang* attribute.

The userInterfaceLanguage tag defines which *xml:lang* value should be selected for labels and help tooltips.

#### 2.1.2 metadataLanguage

Defines the language to be used when retrieving datasets from datasources.

#### 2.1.3 metadataEndpoint

Defines the endpoint of the EDI Server instance that should be used to convert the metadata into its XML format.

#### 2.1.4 sparqlEndpoint

Defines the default SparQL endpoint.

The XSD is yet to be updated with the new data types

This URL is relative to the v3.1 branch: must be updated when the merge to master happens

Could not find specific repo: was it deleted?

### 2.1.5 requiresValidation

Can be set to false (default is true), if you want the metadata to be sent even if they have some errors.

### 2.1.6 baseDocument

This is, as the name suggests, the base of the XML document to be generated: it is a CDATA and it must include the root element, along with any namespaces that need to be defined.

### 2.1.7 xsltChain

A chain of XSL Transformations can be specified to further process the XML generated.

#### Warning

It is the template author's responsibility to make sure they are correct.

## 2.2 endpointTypes

Contains one or more endpointType tags. Each tag defines the interface to communicate with a SparQL endpoint.

It must have these attributes:

Attribute	Description
xml:id	virtuoso or fuseki
method	GET or POST
query	name of the parameter holding the query

And the child tag *parameters*, whose children are *parameter* tags. Each parameter defines a query-string parameter with name and value to be sent to the endpoint.

Name and value are specified as attributes of the *parameter* tags.

## 2.3 datasources

Datasources provide valid values for specific *items*.

A collection of datasources: each datasource can be one of *codelist*, *sparql* or *singleton*.

### 2.3.1 sparql

The most general type of datasource is a SparQL query.

It has two attributes:

Attribute	Description
xml:id	unique id
endpointType	reference to an existing (declared) endpointType

It requires one child tag named *query*, specifying the SparQL query.

Optional child tag *url* allows to override the *sparqlEndpoint* (see 2.1.4) specified in the *settings* tag.

Query can include a *\$search\_param* token, which, if found, will be given a value based, for example, on user text.

### 2.3.2 codelist

A codelist is a simplified version of a *sparql* datasource, based on a pre-defined query, accessed via its URI, specified by the child tag *uri*.

Optional child tag *url* allows to override the *sparqlEndpoint* (see 2.1.4) specified in the *settings* tag.

### 2.3.3 singleton

A singleton is a special stateful *sparql* datasource guaranteed to have only a single instance, so that it can be used to keep some items aligned to some other item whenever the latter changes.

The item triggering said alignment is specified by the attribute *triggerItem*.

Another datasource is always needed, for the singleton to work: the trigger item refers to a *sparql* or *codelist* datasource, whereas the dependent items are connected to it via the singleton, which will refresh and select a single row of the singleton dataset, which is linked, in turn, to the uri of the row selected by the trigger item.

Optional child tag *url* allows to override the *sparqlEndpoint* (see 2.1.4) specified in the *settings* tag.

## 2.4 datasources

Datasources provide valid values for specific *items*.

A collection of datasources: each datasource can be one of *codelist*, *sparql* or *singleton*.

### 2.4.1 sparql

The most general type of datasource is a SparQL query.

It has two attributes:

Attribute	Description
xml:id	unique id
endpointType	reference to an existing (declared) endpointType

It requires one child tag named *query*, specifying the SparQL query.

Optional child tag *url* allows to override the *sparqlEndpoint* (see 2.1.4) specified in the *settings* tag.

Query can include a *\$search\_param* token, which, if found, will be given a value based, for example, on user text.

### 2.4.2 codelist

A codelist is a simplified version of a *sparql* datasource, based on a pre-defined query, accessed via its URI, specified by the child tag *uri*.

Optional child tag *url* allows to override the *sparqlEndpoint* (see 2.1.4) specified in the *settings* tag.

### 2.4.3 singleton

A singleton is a special stateful *sparql* datasource guaranteed to have only a single instance, so that it can be used to keep some items aligned to some other item whenever the latter changes.

The item triggering said alignment is specified by the attribute *triggerItem*.

Another datasource is always needed, for the singleton to work: the trigger item refers to a *sparql* or *codelist* datasource, whereas the dependent items are connected to it via the singleton, which will refresh and select a single row of the singleton dataset, which is linked, in turn, to the uri of the row selected by the trigger item.

Optional child tag *url* allows to override the *sparqlEndpoint* (see 2.1.4) specified in the *settings* tag.

## 3 Data Types

### 3.1 Base data types

#### 3.1.1 text

#### 3.1.2 string

Simplest control type: a small rectangle accepting generic text.

Figure 1: Text example

A text input field with a light gray border. Above the field, on the left, is the text "Other details" followed by a small blue question mark icon. The field itself is empty and has a small cursor at the bottom right corner.

### 3.1.3 URN

Calculated by the server if **isFixed="true"**. Server will generate a valid and unique URN for you.

### 3.1.4 URI

Accepts a string and verifies it's an URI.

### 3.1.5 URL

Accepts a string and verifies it's an URL.

### 3.1.6 int

Accepts a string and verifies it's an only contains numeric digits.

### 3.1.7 float

Accepts a string and verifies it's an only contains numeric digits or the decimal separator (i.e. a dot).

With attribute **show="sliderfloat"**

Shows a slider with a minimum and a maximum value and the position



generates a value for this control.

```

1 <item hasDatatype="float" show="sliderfloat" hasIndex="4"
  xml:id="slider2" isFixed="false" min="0.0" max="100.00"
  step="0.5">
2 <label xml:lang="en">Slider Float 2</label>
3 <label xml:lang="it">Slider Float 2</label>
4 <defaultValue>49</defaultValue>
5 <hasValue>70</hasValue>
6 <hasPath>slider</hasPath>
7 </item>
8

```

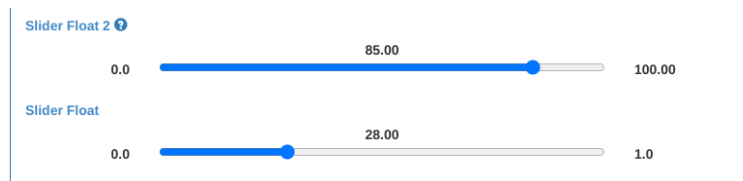


Figure 2: Example of floating-point control with sliderfloat

Shows a slider with a minimum and a maximum value and the position generates a value for this control.

```

1 <item hasDatatype="float" show="sliderfloat" hasIndex="4" xml:id=
  "slider2" isFixed="false" min="0.0" max="100.00" step="0.5">
2 <label xml:lang="en">Slider Float 2</label>
3 <label xml:lang="it">Slider Float 2</label>
4 <defaultValue>49</defaultValue>
5 <hasValue>70</hasValue>
6 <hasPath>slider</hasPath>
7 </item>

```

### 3.1.8 real

Same as *float*.

### 3.1.9 double

Same as *float*.

### 3.1.10 codelist

```

1 <item hasIndex="1" xml:id="ling_md_1" isLanguageNeutral="true"
  isFixed="false" hasDatatype="codelist" datasource="languages"
  show="combobox">
2 <hasPath>/gmd:MD_Metadata/gmd:language/gmd:LanguageCode</hasPath>
3 </item>

```

Still need  
to re-  
implement  
it

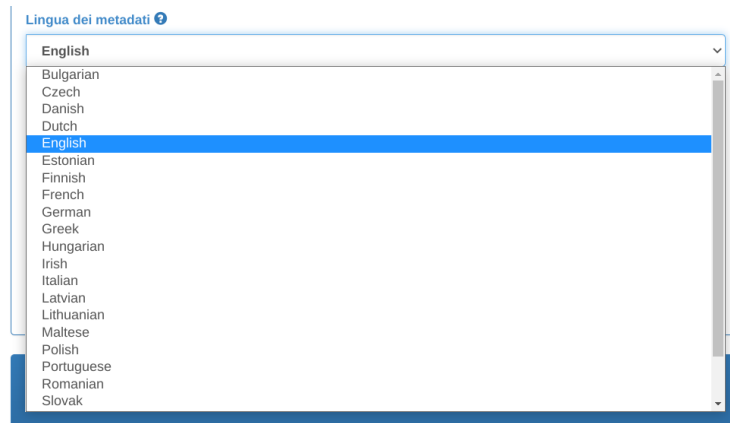
### 3.1.11 autoComplete

Similar to a *codelist*, but preferable with datasources containing many rows.

A textbox querying the datasource associated to the control for matching values. Starts querying when at least 3 characters are entered.

```
1 <item hasIndex="1" xml:id="md_resp_1" outIndex="2" isFixed="false"
  " hasDatatype="autoCompletion" datasource="person">
2 <label xml:lang="en">Email</label>
3 <label xml:lang="it">Email</label>
4 <hasPath>/gmd:MD_Metadata/gmd:contact/gmd:CI_ResponsibleParty/
  gmd:contactInfo/gmd:CI_Contact/gmd:address/gmd:CI_Address/
  gmd:electronicMailAddress/gco:CharacterString</hasPath>
5 </item>
```

Figure 3: Codelist example with *show="combobox"*



### 3.1.12 boolean

Shows a check-box, thus allowing only values **true** or **false**.

## 3.2 Special case data types

### 3.2.1 label

Shows read-only text.

### 3.2.2 image

Given an URL pointing to a valid image in the value, it shows the image.

### 3.2.3 qrcode

Shows whatever the value is as a QR code.

### 3.2.4 select

The select datatype signifies that the item's value is based on some selection occurring in another item called a *trigger item*.

It must be based on a data source of type **singleton** (see 2.4.3).

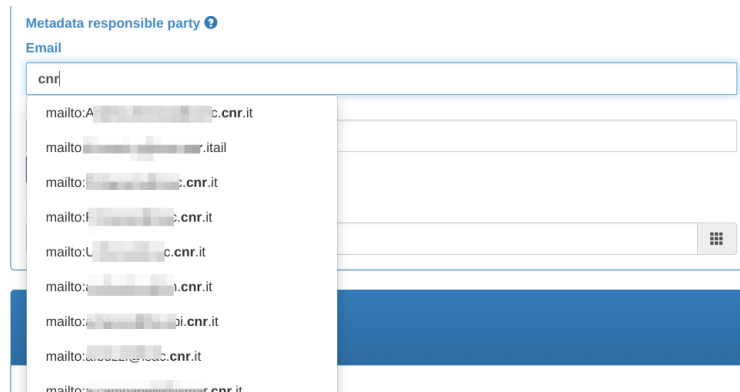
Sometimes the trigger item can be based on the same data source as its connected *select* items, but it can be based on its own data source.

Each *select* item must declare the field it represents in the datasouce.

```
1 <item hasIndex="2" xml:id="resp_2" outIndex="1" field="inst"
  isFixed="false" hasDatatype="select" datasource="personS_2">
2 <label xml:lang="en">Institute</label>
3 <label xml:lang="it">Ente</label>
4 <hasPath>/gmd:MD_Metadata/gmd:identificationInfo/
  gmd:MD_DataIdentification/gmd:citation/gmd:CI_Citation/
  gmd:citedResponsibleParty/gmd:CI_ResponsibleParty/
  gmd:organisationName/gco:CharacterString</hasPath>
5 </item>
```

### 3.2.5 copy

Figure 4: AutoCompletion example



The screenshot shows a web form titled "Metadata responsible party" with a sub-label "Email". Below the label is a text input field containing the text "cni". A dropdown menu is open, displaying a list of email addresses. The first few visible addresses are: "mailto:A...@cni.it", "mailto:...@cni.it", "mailto:...@cni.it", "mailto:f...@cni.it", "mailto:l...@cni.it", "mailto:...@cni.it", "mailto:...@cni.it", "mailto:...@cni.it", and "mailto:...@cni.it". The dropdown menu is styled with a light blue border and a blue header bar.

### 3.2.6 function

Special data type. Its value is calculated by the server by using its template *hasValue* as an XPath run **against the parts of document that have already been generated**.

### 3.2.7 ref

Special data type. In the generated metadata document, it copies the XPath specified in the *hasValue* attribute to the XPath specified by the *hasPath* attribute.

```

1 <item hasDatatype="ref" hasIndex="7" xml:id="title_7" isFixed="
  true">
2 <hasPath>dct:description/@xml:lang</hasPath>
3 <hasValue>/rdf:RDF/dcatapit:Dataset/dct:title/@xml:lang</hasValue
  >
4 </item>

```

In the example above, once the XML document is fully written, the **xml:lang** attribute of **dct:description** is set to equal the same attribute in **/rdf:RDF/dcatapit:Dataset/dct:title**.

### 3.2.8 autonumber

Represents a value that's incremented every time it is encountered within the containing element.

Value is assigned by EDI Server.

### 3.2.9 hidden

Hidden item: it is *fixed* by default (i.e. read-only).

### 3.2.10 date

Requests a date from the user, via a small calendar.

Requires a *defaultValue*, which can be the macro *\$TODAY\$*.

```

1 <item hasDatatype="date" hasIndex="7" xml:id="test_7" isFixed="
2   false">
3 <label xml:lang="en">Issues date</label>
4 <label xml:lang="it">Data di rilascio</label>
5 <help xml:lang="en">Help</help>
6 <help xml:lang="it">Help</help>
7 <hasValue>dct:issued</hasValue>
8 <defaultValue>$TODAY$</defaultValue>
9 </item>
10

```

Figure 5: String example, in this case it is part of a *isMultiple="true"* element, as you can tell from the "+" button underneath it

### 3.2.11 dateRange

Same as *date*, except that it requests a start and an end date.

```

1
2 <item hasIndex="8" xml:id="est_temp_8" isFixed="false"
  hasDatatype="dateRange">
3 <label xml:lang="en">Start date</label>
4 <label xml:lang="it">Data inizio</label>
5 <start>
6 <label xml:lang="en">Start date</label>
7 <label xml:lang="it">Data inizio</label>
8 <hasPath>/gmd:MD_Metadata/gmd:identificationInfo/
  gmd:MD_DataIdentification/gmd:extent/gmd:EX_Extent/
  gmd:temporalElement/gmd:EX_TemporalExtent/gmd:extent/
  gml:TimePeriod/gml:beginPosition</hasPath>
9 </start>
10 <end>
11 <label xml:lang="en">End date</label>
12 <label xml:lang="it">Data fine</label>
13 <hasPath>/gmd:MD_Metadata/gmd:identificationInfo/
  gmd:MD_DataIdentification/gmd:extent/gmd:EX_Extent/
  gmd:temporalElement/gmd:EX_TemporalExtent/gmd:extent/
  gml:TimePeriod/gml:endPosition</hasPath>
14 </end>
15 </item>
16

```

### 3.2.12 boundingBox

Requests a geographic bounding box from the user. It can be specified either by inputting the coordinates in 4 text boxes, or by drawing a rectangle on a map.

```

1
2 <item hasIndex="1" xml:id="loc_geo_1" isFixed="false" hasDatatype
  ="boundingBox">
3 <westLongitude outIndex="1" queryStringParameter="westlon">
4 <label xml:lang="en">W longitude</label>
5 <label xml:lang="it">Longitudine 0</label>
6 <hasPath>/gmd:MD_Metadata/gmd:identificationInfo/
  gmd:MD_DataIdentification/gmd:extent/gmd:EX_Extent/
  gmd:geographicElement/gmd:EX_GeographicBoundingBox/
  gmd:westBoundLongitude/gco:Decimal</hasPath>
7 </westLongitude>
8 <eastLongitude outIndex="2" queryStringParameter="eastlon">
9 <label xml:lang="en">E longitude</label>
10 <label xml:lang="it">Longitudine E</label>
11 <hasPath>/gmd:MD_Metadata/gmd:identificationInfo/
  gmd:MD_DataIdentification/gmd:extent/gmd:EX_Extent/
  gmd:geographicElement/gmd:EX_GeographicBoundingBox/
  gmd:eastBoundLongitude/gco:Decimal</hasPath>
12 </eastLongitude>
13 <northLatitude outIndex="4" queryStringParameter="northlat">
14 <label xml:lang="en">N latitude</label>
15 <label xml:lang="it">Latitudine N</label>
16 <hasPath>/gmd:MD_Metadata/gmd:identificationInfo/
  gmd:MD_DataIdentification/gmd:extent/gmd:EX_Extent/
  gmd:geographicElement/gmd:EX_GeographicBoundingBox/
  gmd:northBoundLatitude/gco:Decimal</hasPath>
17 </northLatitude>

```

```

18 <southLatitude outIndex="3" queryStringParameter="southlat">
19 <label xml:lang="en">S latitude</label>
20 <label xml:lang="it">Latitudine S</label>
21 <hasPath>/gmd:MD_Metadata/gmd:identificationInfo/
    gmd:MD_DataIdentification/gmd:extent/gmd:EX_Extent/
    gmd:geographicElement/gmd:EX_GeographicBoundingBox/
    gmd:southBoundLatitude/gco:Decimal</hasPath>
22 </southLatitude>
23 </item>

```

Figure 6: Date example

Issues date ⓘ

2023-06-14

« June 2023 »

Su	Mo	Tu	We	Th	Fr	Sa
28	29	30	31	1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	1
2	3	4	5	6	7	8

1.00

1.0

60.50

100.00

## Todo list

■ The XSD is yet to be updated with the new data types . . . . .	4
■ This URL is relative to the v3.1 branch: must be updated when the merge to master happens . . . . .	4
■ Could not find specific repo: was it deleted? . . . . .	4
■ Still need to reimplement it . . . . .	8