10TH EXPERT GROUP MEETING ON

Statistical Data and Metadata eXchange

JANUARY 25-28, 2021

Improved Codelists' Handling

January 27, 2021

Edgardo Greising
Head of Knowledge Management Solutions Unit / ILO
Chair SDMX-TWG

sdmx
Statistical Data and Metadata eXchange

Issues with Codelists

- A Concept can be enumerated by one and only one Codelist
- Using Global Codelists (cross-domain or domain specific) is a good practice
- Usually, an Organization using such Codelists needs to add some own codes
  - e.g. National codes (5th digit) in ISIC Rev.4
- In other cases, they want to use the harmonized Codelists, but don't need all the items
  - e.g. Only a sub-set of all the age bands in IAEG-SDGs:CL_AGE(1.0)
- ***The only solution is to create this new Codelist 'by hand', selecting the items to include from the global Codelist and/or adding the new items.***

Solution

- Enable a Codelist to be extended in order to include the Codes from other Codelists.
  - Resolution of duplicates
    - included codes can either be given a sequence, or a unique prefix defined
  - Including a explicit subset of codes from the other Codelists
    - specific lists of codes may be defined for either inclusion or exclusion
  - Expressions defining which codes to include
    - the '%' wildcard may be used in a similar way to Constraints
  - Exchange of either the resulting 'resolved' Codelist, or a 'raw' description of how it is composed

Solution explained

- A Codelist can extend one or more Codelists.
  - Codelist extensions are defined as a list of references to parent Codelists.
- When two codelists have items with the same Code Id, the Codelist referenced later takes priority.
  - The '**sequence'** may be used to establish the order that will be used when extending a Codelist
  - As the extended Codelist may also define its own Codes, these take the ultimate priority over any extension Codelists.

Solution explained

- A reference to a Codelist may contain a prefix.
  - This '**prefix'** will be applied to all the codes in the Codelist before they are imported into the extended Codelist.
- An explicit list of Code Ids may be provided for explicit inclusion or exclusion.
  - May contain '**wildcards'** using the same notation as Constraints (%).
  - **'Cascading'** values is also supported using the same syntax as the Constraints.
  - It is also possible to include children and exclude the Code by using '**excluderoot'**
  - Exclusion and inclusion is not supported against a single Codelist.

Example

```
<?xml version="1.0" encoding="UTF-8"?>
<mes:Structures>
    <str:Codelists>
        <str:Codelist agencyID="ILO" id="CL_BASE" version="1.0">
            <com:Name xml:lang="en">Base Code List</com:Name>
            <str:Code id="CODE1">
                <com:Name xml:lang="en">Code 1</com:Name>
            </str:Code>
            <str:Code id="CODE2">
                <com:Name xml:lang="en">Code 2</com:Name>
            </str:Code>
        <str:Extendedby>
            <Ref>
                <Agency>BIS</Agency>
                <Id>CL_REF_AREA</Id>
                <Version>1.0</Version>
            </Ref>
            <CodeSelection isIncluded="true">
                <Value cascadeValues="true">UK</Value>
                <Value>FR</Value>
                <Value>DE</Value>
            </CodeSelection>
        </str:Extendedby>
        <str:Extendedby>
            <Ref>
                <Agency>UNICEF</Agency>
                <Id>CL_CONTINENT_BREAKDOWN</Id>
                <Version>2.0</Version>
            </Ref>
            <CodeSelection>
            <Value cascadeValues="true"></Value>
            </CodeSelection>
        </str:Extendedby>
        </str:Codelist>
    </str:Codelists>
</mes:Structures>
```

The flagship use case: Discriminated Union

- Code Lists representing breakdowns may frequently include several variants of the classification. For example, the standard classification of economic activities (ISIC) includes several revisions, plus aggregations; each of them is a variant.
- These variants are mutually exclusive, in the sense that, although they enumerate the same concept, only one should be used at a time, based on certain context: country, time reference, representativeness of the sample, etc.
- In SDMX, the “context” is defined at the Dataflow or Provision Agreement level.

Issues with multiple variants

- A single Code List can be defined as the representation of the concept “ACTIVITY”, which must include all the categories for all the variants that may be used, i.e. ISIC Rev. 4 codes, plus Rev. 3.1 codes, and any aggregate(s) used by the particular implementation.
- The result is a ***huge code list, hard to maintain***, for which only a small percentage of the codes are relevant for each Dataflow.
- A Dataflow would reference a “generic” DSD for all data reporters, but depending on the context of each of them, different code sets (i.e. different variants) should be used.
- Since only one Code List enumerates the “ACTIVITY” concept, a Constraint should be defined for each dataflow to use a particular variant.
- In other words***, it is required to have one dataflow with a specific constraint per variant used to select the proper codes***.

Solution: Discriminated Union

- Two issues to solve:
  1. **the burden of maintaining a huge Code List** with all the variants
  2. the selection of a **different subset of codes depending on the PA**
- Having ***independent Code Lists for each variant*** (i.e.each classification version) *solves issue 1*.
- Have the Dimension ACTIVITY represented by the Code List ***CL_ACTIVITY with no codes***
- CL_ACTIVITY has ***extension references*** to CL_ISIC4, CL_ISIC3, CL_AGGR, etc..
- In the extension clause, ***a "prefix" attribute*** is specified for each one, as ISIC4_, ISIC3_, AGGR_, etc..
- ***Each PA has a specific ContentConstraint*** to keep the items of the variant used by the data provider (*solves issue 2)*

How it works:

- Each variant in a separate Code List facilitates the maintenance and allow keeping the original codes, regardless of potential conflicts:
  - ISIC Rev. 4: “A” represents “Agriculture, forestry and fishing”,
  - ISIC 3.1: “A” means “Agriculture, hunting and forestry”
- Specifying “***prefix=<variant_>”*** for each Code List in the “***ExtendedBy***” clause prevents duplicates
  - CL_ISIC4 with ***prefix=“ISIC4_”*** gets “ISIC4_A”
  - CL_ISIC3 with ***prefix=“ISIC3_”*** returns “ISIC3_A”.

How it works:

- Each PA has a specific ContentConstraint to include ***Value="<variant>_%"*** items and ***removePrefix="<variant>_"***
- A query for the ***PA*** with ***references=descendants*** and ***detail=referencepartial*** will return CL_ACTIVITY with the extensions resolved and the constraints applied, so ***it will only include codes originally from CL_<variant>***.