CLARIN
Common Language Resources and Technology Infrastructure

CMDI 1.2 Schema Sanity WG Status Report

Menzo Windhouwer & Oliver Schonefeld
The Language Archive - DANS & IDS Mannheim
menzo.windhouwer@dans.knaw.nl & schonefeld@ids-mannheim.de

CMDI Taskforce Meeting
Utrecht
2014-02-21

Issues

- General component schema consistency
- Namespace per profile

Issue 1: General consistency

Problem:

“Multiple developers worked on the *general component schema* and they made different choices in XML schema design, e.g., what should be an element and what should be an attribute, what should be capitalized or not.”

Issue 1: General consistency

- Common for all solutions
  - Make a choice between attributes and elements, or allow a blend. Also cleanup names in the process, i.e., to capitalized.
- Three different “flavors”
  - Solution 1: “metadata in attributes, data in elements”
  - Solution 2: “elements only”
  - Solution 3: “attributes only”

Issue 1: General consistency

- Proposed solution
  - Solution 3 (“attributes only”)
- Centre Impact
  - *No* impact on CMDI metadata instances
  - Tools that use XML specification of a profile or component need to be changed
    - That’s mostly CMD Infrastructure tools itself
- Open Issues/Questions/Notes
  - Not all information can be strictly “attribute only”, e.g. multi-lingual documentation of profiles / components
  - Also clean up identifiers and prune “CMD_” prefix

Issue 2: Namespace per profile

Problem (1/3):

"The metadata in each record returned by ListRecords and GetRecord **must** comply with the conventions of the XML namespace specification. This means that the root element of the metadata part **must** contain an xmlns attribute, the value of which is the XML namespace URI of the metadata format. The root element **must** also contain an xsi:schemaLocation attribute that has a value that includes the URL of the XML schema for validation of the metadata. This URL **must** match the URL of the metadata schema for the metadataPrefix included as an argument to the ListRecords or GetRecord request (the mapping from metadataPrefix to metadata schema is defined by the repository's response to the ListMetadataFormats request)." (**3.4 metadataPrefix and Metadata Schema**)

Issue 2: Namespace per profile

Problem (2/3):

"We envision applications of Extensible Markup Language (XML) where a single XML document may contain elements and attributes (here referred to as a "markup vocabulary") that are defined for and used by multiple software modules. [..]
Such documents, containing multiple markup vocabularies, pose problems of recognition and collision. Software modules need to be able to recognize the elements and attributes which they are designed to process, even in the face of "collisions" occurring when markup intended for some other software package uses the same element name or attribute name.“
(Namespaces in XML 1.0, Section 1 "Motivation and Summary“)

Issue 2: Namespace per profile

Problem (3/3):

- In other words, a single value of metadataPrefix can only correspond to a single value of xsi:schemaLocation.
- However, each CMDI profile has it's own xsi:schemaLocation but they is only one metadataPrefix
- CMDI has a rather *idiosyncratic* interpretation of the XML Namespace spec, because a **single** XML namespace name refers to **multiple sets** of vocabularies (= profile) with possibly **conflicting** element definitions

Issue 2: Namespace per profile

Potential Problems with the “all-in-one” Namespace approach:

- An **XML Parsers** parsing and validating a batch of CMDI instances from various profiles can cache an internal representation of a parsed XML schema based on XML Namespaces to speed up the processing.
- For example, Xerces-J has the following comment in org.apache.xerces.impl.xs.XMLSchemaValidator.java:1564 ff

  "store the external schema locations. they are set when reset is called, so any other schemaLocation declaration for the same namespace will be effectively ignored. because we choose to take first location hint available for a particular namespace.”

  - The comment is in the reset() method of the SchemaValidator. This hints, that the parser might cache schemas based on XML Namespace names, if instances of the parser get re-used.

Issue 2: Namespace per profile

Potential Problems with the “all-in-one” Namespace approach:

- **XML Databases**: (especially) with validation turned on, native XML databases might also run into the XML schema caching issue or have trouble locating the appropriate XML schema for a CMDI instance.
- NB:
    - The popular XML database eXist-db uses Xerces-J under the hood.

Issue 2: Namespace per profile

Possible Solutions (1/3):

1. “Be pragmatic” (and leave things as they are)
2. “Profile specific metadataPrefix”
3. “Leave it up to the centers”
4. “CMD envelope namespace and (one) payload namespace”
5. “CMD envelope namespace and profile specific namespaces”

Issue 2: Namespace per profile

Possible Solutions (2/3):

1. "Be pragmatic" (and leave things as they are)
   - Violates OAI standard; CLARIN is about standards
2. "Profile specific metadataPrefix"
   - Idiosyncratic use of OAI
     - CMDI 1.2 should do better and not introduce new quirks
   - NB: currently recommended work-around for CMDI 1.1
3. "Leave it up to the centers"
   - Will lead to fragmented center landscape
   - CMDI exploitation will require center specific quirks

Issue 2: Namespace per profile

Possible Solutions (3/3):

4. “CMD envelope namespace and (one) payload namespace”
   - Generally better solution than 1-3
   - Solves OAI conformance (with minor quirk of cmdi-minimal.xsd)
   - … but leaves the other “XML problems” unaddressed
5. “CMD envelope namespace and profile specific namespaces”
   - Cleanest solution
   - … but also most painful ☹

Issue 2: Namespace per profile

- Proposed solution
  - “CMD envelope namespace and profile specific namespaces”
- Centre Impact
  - All tools that work with CMD records need to be changed
  - All CMD records need to be changed

Issue 2: Namespace per profile

- The following approaches could be used to *lessen the impact* for those, who don’t care or don’t want so deal with XML Namespaces
  - ignore Namespaces in XPath:
    - in XPath 1.0: *[local-name()='ToolService’]
    - in XPath 2.0: *:ToolService"
  - use a "flattener XSLT”
    - A XSLT-stylesheet, that puts all elements into the same namespace (and deliberately breaking validation). Users can than use their traditional XPath expressions. However, "flattened" CMDI instances **must** only be used in transient contexts, i.e. they **must not** be used for long term storage or exchange.
  - use a *SAX filter* to change incoming namespaces
- When converting CMD records at a center, a *conversion script* could try handle the Namespace stuff (tbd)

Issue 2: Namespace per profile

```
<cmd-e:CMD
  xmlns:cmd-e="http://www.clarin.eu/cmd/envelope"
  xmlns:cmd-p="http://www.clarin.eu/cmd/payload/p12345"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.clarin.eu/cmd/envelope
                      http://infra.clarin.eu/cmd/xsd/minimal-cmdi.xsd
                      http://www.clarin.eu/cmd/payload/p12345
                      http://catalog.clarin.eu/.../profiles/p12345/xsd"
  CMDVersion="1.2">
  <cmd-e:Header>
    ...
  </cmd-e:Header>
  ...
  <cmd-e:Components>
    <cmd-p:ToolService>
      ...
    </cmd-p:ToolService>
  </cmd-e:Components>
</cmd-e:CMD>
```