DSpaceIntermediateMetadata

Contents

DSpace Intermediate Metadata Format

This page proposes an XML notation for DSpace's internal Item metadata, that is, the metadata fields stored in the database for each Item. It is used by XsltCrosswalk. It is called the *Intermediate* format because it is inteded **solely** as an intermediate stage in XML-translation-based crosswalks. To reiterate, **This is an INTERMEDIATE format, it is NOT for exporting or harvesting metadata!**

XSLT translation

Extensible Stylesheet Language (XSL) and XSLT is a powerful mechanism for transforming one XML expression into another. Since many metadata formats are expressed in XML, you can use XSL stylesheets quite effectively to implement crosswalks.

See the XsltCrosswalk for an example of how to do this.

However, you have to start with (or end up with) an XML document. That is why we need the DSpace Intermediate Metadata format. To generate XML metadata *from* an Item, the steps are:

1. Generate this Intermediate XML format directly from the item.
2. Invoke an XSLT engine and your XSL stylesheet to translate that to the target metadata format, e.g. MODS.

On submission, the steps are reversed:

1. Invoke an XSLT engine and your XSL stylesheet to translate the incoming metadata format to DSpace Intermediate Format.
2. Hand the intermediate XML to a method that stuffs its field values into the Item (like many calls to `Item.addDC()`).

The Format

It cannot be overemphasized that this is strictly an **internal** metadata format. It must never be recorded, transmitted, or exposed outside of DSpace, because it is **NOT** any sort of standard metadata. We must not allow it to "escape" to prevent its being mistaken for an actual supported and sanctioned metadata format. It exists to support internal transformations *only*.

This format is designed to be a straightforward and precise representation of the DSpace data model's Item metadata. It represents the new metadata model described in MetadataSupport, which includes a "metadata schema" field.

Namespace

See XmlNamespaces for details. All elements are in the "dim" namespace, identified by the URI `http://www.dspace.org/xmlns/dspace/dim`

There will eventually be a schema for this namespace, as soon as `dspace.org` establishes a place to put schemas. The purpose of the schema is to document the {dim}} element and allow validation.

Elements

Here is an example of a metadata record:

```
<dim:dim xmlns:dim="http://www.dspace.org/xmlns/dspace/dim" dspaceType="ITEM">
  <dim:field mdschema="dc" element="title" lang="en_US">
    The Endochronic Properties of Resublimated Thiotimonline
  </dim:field>
  <dim:field mdschema="dc" element="contributor" qualifier="author">
    Isaac Asimov
  </dim:field>
  <dim:field mdschema="dc" element="language" qualifier="iso">
    eng
  </dim:field>
  <dim:field mdschema="dc" element="subject" qualifier="other" lang="en_US">
    time-travel scifi hoax
  </dim:field>
  <dim:field element="publisher">
    Boston University Department of Biochemistry
  </dim:field>
</dim:dim>
```

The root element is named `"dim"` (for DSpace Intermediate Metadata, also because it is an unappealing name in English to discourage exposing it!). This element may contain one attribute (along with namespace declarations):

- The `dspaceType` attribute is the type of dspace object being described. The possible values of this attribute are: "`ITEM`", "`COLLECTION`", or "`COMMUNITY`".
- The `dim` element contains a list of 0 or more `"field"` elements, each of which describes a single value. In each `field` element:
- The `mdschema` attribute is the metadata schema, aka "namespace", described in MetadataSupport. In this example all fields are "dc", meaning the original DSpace LAP qualified DC. This could be the default when that attribute is omitted.
- The `element` attribute is the Dublin Core element name, or its equivalent in another schema. *It is required.*
- The `qualifier` attribute is the DC qualifier or equivalent. Omitting it means the qualifier is null.
- Finally, the `lang` attribute is the language code associated with the entry. I deliberately did not use the XML standard `xml:lang` name for this attribute because it implies semantics that we cannot guarantee to support, since the value of this attribute is whatever someone put into DSpace.
- The text value of the `field` element is the value of the metadata field.
- Any number of `field` elements are allowed, even with all attributes matching.

An XML Schema document (XSD) description will be forthcoming once this design is approved.

Extensions

For ItemBatchUpdate (importing existing bibliographic data and uploading of corresponding files), DIM is "extended" the following ways:

1. `<dim:list>...</dim:list>` now can enclose multiple items: `<dim:dim>...</dim:dim>`
2. `<dim:field ... type="field-type">...</dim:field>` is used to specify either *type="unique"* (to remove prior field content before inserting new one) either *type="key"* (to specify that the current record replaces the record which may (or not) exist with the same value: useful for external identifiers)
3. `<dim:remove mdschema="schema-name" element="element-name" qualifier="qualifier-name" lang="language_country" />` remove field occurrence(s) corresponding to the specified *element-name*, *qualifier-name* and (optional) *language_country*.
4. `<dim:original>file-path</dim:original>` specifies the path of the document file to upload (not a "symbolic link")
5. `<dim:licence>file-path</dim:licence>` specifies the path to the licence file to upload
6. `<dim:collection> collection complete handle or internal number </dim:collection>` : Additional collection to link with the document

Modifications are in `XSLTIngestionCrosswalk` and are therefore common to `ItemImport` and `XSLTingest` (described in ItemBatchUpdate).

Opinions? Comments?

Should DIM support the types of BITSTREAM and/or BUNDLE? If so what fields would be used? – ScottPhillips