# ItemBatchUpdate

## Motivation

One often wants to import/update metadata or documents from external sources.

XSLT is already used in DSpace to establish "crosswalks" between one data representation to another: XsltCrosswalk

An internal metadata representation called DIM is standardized: DspaceIntermediateMetadata

Christophe.Dupriez 10:33, 10 August 2007 (EDT)

## Work done

An ingestion module *XSLTingest* has been developped which can import an XML file for which an XSLT crosswalk has been configured: this XML file can contain multiple bibliographic records which are then all loaded automatically. *I propose to rename it ItemBatchUpdate before committing*.

An XSLT crosswalk from Pubmed (MedLine) XML format to an DSpace "overloaded DC" metadata structure has been developped.

This XSLT uses a Java function for Medline date conversion (I did not succeed to use JavaScript up to now): the corresponding compiled java class must therefore be included in the CLASSPATH.

This can be found on SourceForge

## Configuration

You first have to define a crosswalk in *dspace.cfg* like it is explained in XsltCrosswalk. For example:

crosswalk.submission.PUBMED.stylesheet= crosswalks/pubmed-submission.xsl

XSLT is better explained further below.

## Batch Command Line

You simply use a DSpace Batch function (with dsrun) with similar parameters than ItemImport, except for two:

- -s or --source is a file and not a directory (multiple XML records in one file)
- -x or --crosswalk name provide the name of the XSLT crosswalk to use (XSLT Submission Crosswalks must be declared in dspace.cfg).

~~a (~~-add) is used for importing new items AND for updating existing items for which the handles are NOT specified in the map file.

Example under Windows:

```
call D:\DSPACE\BIN\DSRUN org.dspace.content.crosswalk.XSLTingest
-a -x PUBMED -c 443803704/3 -s source1.xml -m source1.map -e christophe.dupriez@destin.be
```

All the functionalities around the "mapfile" are kept (resume, delete).

The source file is translated into DIM using the XSLT template.
The generated DIM is then imported (or submitted if --workflow is specified).

## Writing an XSLT Crosswalk

The XSLT will generate DIM XML. It should start by:

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:dim="http://www.dspace.org/xmlns/dspace/dim"
xmlns:pubmed="http://www.ncbi.nlm.nih.gov/entrez/query/DTD/pubmed_070101.dtd"
"(or any other XMLNS of the source file)"
xmlns:java="http://xml.apache.org/xalan/java"
version="1.0">
<xsl:output indent="yes" method="xml"/>
```

Each input record should be matched and produce:

```
<xsl:element name="dim:dim">
<xsl:element name="dim:field">
<xsl:attribute name="mdschema">dc</xsl:attribute>
<xsl:attribute name="element">identifier</xsl:attribute>
<xsl:attribute name="qualifier">pmid</xsl:attribute>
"(or any other field used to store an external identifer of the record)"
<xsl:attribute name="type">key</xsl:attribute>
<xsl:value-of select="normalize-space(PMID)"/>
"(or any other element of input file)"
</xsl:element>
<xsl:element name="dim:remove">
<xsl:attribute name="mdschema">dc</xsl:attribute>
<xsl:attribute name="element">contributor</xsl:attribute>
<xsl:attribute name="qualifier">author</xsl:attribute>
</xsl:element>
<xsl:element name="dim:remove">
<xsl:attribute name="mdschema">dc</xsl:attribute>
<xsl:attribute name="element">subject</xsl:attribute>
<xsl:attribute name="qualifier">mesh</xsl:attribute>
</xsl:element>
... You have to remove all multi-valued field that you completely reimport ...

... This test if an associated document has to be uploaded and
generates a <dim:original...> if necessary
<xsl:variable name="linkedFile"
select="java:org.dspace.content.crosswalk.XwalkHelper.checkFile
('crosswalk.submission.PUBMED.documents',PMID)"/>
<xsl:if test="string-length($linkedFile) > 0">
<xsl:element name="dim:original">
<xsl:attribute name="type">unique</xsl:attribute>
<xsl:value-of select="normalize-space($linkedFile)"/>
</xsl:element>
</xsl:if>

... other templates are defined for each individual field
<xsl:apply-templates/>

</xsl:element>
```

You can add parameters for your XSLT program:

```
crosswalk.submission.PUBMED.documents= Z:\\downloaded\\pmid\\[id].PDF
```

If a field is never repeated:

```
<xsl:element name="dim:field">
<xsl:attribute name="mdschema">dc</xsl:attribute>
<xsl:attribute name="element">identifier</xsl:attribute>
<xsl:attribute name="qualifier">issn</xsl:attribute>
<xsl:attribute name="type">unique</xsl:attribute>
<xsl:value-of select="normalize-space(ISSN)"/>
</xsl:element>
```