

Topic

• A Introduction

- This is a temporary repository for some questions that have arisen so far in the initial round of the project. We hope to get this into a wiki or other more satisfactory format soon.

• Contents:

B Policy questions

- Security concerns re COBOAT interfacing to TMS
- How to handle Creator if you don't have one?

C Installation and set up

- How long does it take to run?
- What platforms can I run this on?
- I want to restrict the objects covered by the export

D Basic configuration questions

- How can we have different CDWA Lite based on department
- What logic should I put in retrieve, in first pass, in second pass?
- I want to change part of the output
- How do I define OAI-PMH sets?

E Detailed programming questions

- Do joins in "First Pass" scripts behave like inner or outer joins?
- Is there any way to do further text processing on fields?

F Does this look right? How can I debug this nightmare?

- Does this database look right...?
- I am looking for sample queries to run against the set?
- Is there an easy way to figure out what's going into arrays?
- I keep getting error messages "Unknown Array"....

• B Policy questions

• 1 Security concerns re COBOAT interfacing to TMS

- Q: The COBOAT application will be interfacing directly with TMS and will have, theoretically, unfettered access to everything in TMS. This can be limited via ODBC to avoid altering data but still the program could retrieve sensitive data from TMS.

Will the COBOAT application pull everything out of TMS and then sort it out (which would be problematic)

OR

Will COBOAT only pull from TMS via the configuration mapping file what is needed to create the valid CDWA Lite record? How can this be verified / audited?

- A: COBOAT accesses the TMS database via ODBC, using a specific account. This account has to be set by the institution, and it is strongly recommended that a username is set up specifically for this purpose, that has access only to this database, and is limited to read access only.

The data that COBOAT retrieves is configured in the TMSretrieve.cbcs script, which is under institution control. The actual select statements issued by COBOAT can be seen in the console and are recorded in the service/sourcedata/TMS/cbxrep.txt retrieval log written for each run.

Within the TMSretrieve.cbcs script, the institution can configure the tables, and the columns within each table, that are retrieved; and can restrict these to certain records using 'where' clauses.

Within SQL Server, the database administrator can limit the 'COBOAT' login account to access only certain tables of the TMS database; and if further security is required, it would be possible to design database views that included only desired tables, columns, and even records, and grant access to this view instead of to the database.

• 2 How to handle Creator if you don't have one?

- Q: <cdwalite:displayCreator> is a required element in CDWA Lite, but many of our records do not have a creator assigned in TMS. How should we handle this?

• D(iscussion):

Where an object is not the work of one or more identifiable individuals, different institutions have adopted different ways of representing this in TMS.

The default configuration scripts for MDEP populate the <cdwalite:displayCreator> element with DisplayName of relevant TMS constituents, concatenated with suitable delimiters. If there are no such constituents, the <cdwalite:displayCreator> element is included but is empty; and a warning specifying the affected object is included in the 'displayCreator' record.

Some institutions have modified this to add "n/a" as the value of the <cdwalite:displayCreator> element (and still report the case) when there are no relevant constituents, in order to be compliant with at least the letter of the CDWA Lite specification. Others have modified the script to instead fall back on the Culture and Period associated with the object, so that this element will be populated with something that more closely matches the intent of a displayable attribution string (and report these cases separately, while still reporting in a separate file those cases where even the Culture and Period cannot be found).

How best to handle this case will depend mainly on how your institution deals with these issues, and to some extent is a matter of debate within the community. You may get some more information on this from the mailing list at <http://listserv.oclc.org/archives/data-exchange-tools-l.html>

• C Installation and set up

• 1 How long does it take to run?

Q: I'm looking to get a sense of how long it will take to run the first big extract of our 200,000 records. Is it going to be hours or days?

- A: Based on experiences so far the whole process can take between 20 minutes and six hours. In addition to the number of records, the variables include the speed of the server hosting the TMS database, the server hosting the OAI/Cat/Museum database, the machine that COBOAT is running off, and the network connections between them.

• 2 What platforms can I run this on?

- Q: Does it have to run off a PC? I thought (and I could easily be wrong on this) that COBOAT could be run on a Mac ...

• A: COBOAT runs on Mac or Windows. Both are on the download site, just download the one you want.

- Q: Does it run on Linux?

• A: Cogapp have a version of COBOAT running on Linux, which has yet to be deployed at a client site. We have encountered issues with ODBC drivers retrieving data from TMS/SQL Server, but these have not been thoroughly worked through and may have been specific to particular test setups. If you would like to run this on Linux, please contact <coboat@cogapp.com> to discuss further.

• 3 I want to restrict the objects covered by the export

- The scope of the generation - the set of objects for which CDWA Lite records will be produced - can be set using a pair of attributes to select a column of object IDs from a tab delimited file in the source data directory. In the simplest (and default) case, this example generates records for every object retrieved from TMS:

```
ScopeSrcFile="TMS/Objects.txt"
ScopeSrcColumn="ObjectID"
```

So the simple way to restrict the objects for which CDWA Lite records will be produced is to add a 'where' clause to the element in TMSretrieve.cbcs which retrieves the basic data from the Objects table.

If you have some other or more complicated way to define the set of objects for which you wish to generate CDWA Lite records, then you can leave the objects table alone, and create a separate file to set scope. For example, if you wanted to use a TMS object package to define the scope, you could add an element to TMSretrieve.cbcs to extract this:

Topic

```
<table destfile="ScopeFile.txt" sourcetable="ObjectPackages_ObjPkgList"
  where="(ObjectPackages.ObjectPackageID = ObjPkgList.ObjectPackageID) AND (ObjectPackages.Name = 'CDWALiteExport')">
  <field name="ObjectID" />
</table>
```

and modify the parameters in the master script accordingly:

```
ScopeSrcFile="TMS/ScopeFile.txt"
ScopeSrcColumn="ObjectID"
```

Either approach will produce the same result, in terms of the records that are rendered and sent to the OAI CatMuseum database. The advantage of the first approach is that it will take (a bit) less time to do the retrieve, and consume (a bit) less disk space; the advantage of the second approach, if you expect that you might subsequently extend the scope of the CDWA Lite/OAI coverage, is that all the data is there for you to experiment with.

Also note that the ScopeSrcFile needn't be generated out of TMS: it could be from another database, or hand edited (or custom extracted) and stored in some other sub-folder of the 'sourcedata' folder.

- Q: I want to restrict the objects covered by the export, so I added a filter in first_pass.xml file - why didn't that work?

I have modified TMSretrieve.cbcs to add

```
<field name="ObjectStatusID"/>
<field name="IsTemplate"/>
```

to the Objects table tag (i.e. moved them up from the commented fields), and modified first_pass.xml to change the Objects array to:

```
<array name="Objects" table="TMS.Objects" link="ObjectID">
  <filter>{TMS.Objects.IsTemplate}=0 and {TMS.Objects.ObjectStatusID}=1</filter>
</array>
```

why didn't this work?

- A: See above for the correct way to restrict the set of objects covered by the export. The key thing to understand is that the 'first pass' and 'second pass' processing are operated successively on each object 'in scope'. So anything that goes in first_pass.xml is too late to affect scope - at that point COBOAT is already committed to generating a CDWA Lite record for this object.

- D Basic configuration questions

- 1 Q: How can we have different maps into CDWA Lite based on the curatorial department of the record?

- A: If the differences are minor, this is probably best dealt with in the second pass configuration, using the Smarty syntax logic to select between different fragments of CDWA Lite or Smarty syntax to output.

In some cases you might want to handle a difference in the first pass, loading different data into an array, or creating two different arrays, one for each case.

It would be possible to run completely separate passes for each department (with different retrieve, first pass and second pass configuration scripts) : in this case you would need two versions of the build script, the default one with mode="replace", and a second one with mode="append". Then the master script would run the first retrieve, munge and build; followed by the next one; and so on. Provided that the object ids, and any relevant set ids, were distinct, this should work (but is probably overkill unless the differences are really very dramatic).

- 2 What logic should I put in retrieve, in first pass, in second pass?

- Q: One thing that's been difficult to resolve is where to make use of the different record filters we need to apply. It seems to make sense to have some broader filters in the TMSretrieve.cbcs script (for example excluding entire departments and deaccessioned objects from the total set of objects) and more specific filters in the first_pass.xml script. I'm not sure what makes the most sense. What are others doing?

- A: First, it's important to understand that the first pass and second pass are processed in turn for each object in the scope of the run; and it may be easiest to think about this backward from the second pass.

For each object, COBOAT loads data into Smarty arrays, and then expands second_pass.tpl to render the CDWA Lite record for that object. The Smarty tags in the second pass cannot reference the data retrieved from TMS directly, it can only reference the arrays specified in first_pass.xml. So the purpose of the first pass is to load the data from TMS, related to the particular object being processed, into arrays to make it available to the second pass.

One way to deal with this is to make a minimal first pass file which simply loads each retrieved table into a corresponding array, eg

```
<array name="Geography" table="TMS.ObjGeography" link="ObjectID" />
```

Then for all records directly linked to a particular object, all the retrieved data will be directly accessible to the Smarty tags in the second pass file.

Why do more than that? Mainly to make the second pass much easier to read (and compensate for the limitations of the Smarty emulation). For example, we might be going to loop over the primary titles for an object. TMS may store many titles for a given object: primary, secondary, old, original, foreign language, etc. If the first pass has a declaration like this:

```
<array name="Titles" table="TMS.ObjTitles" link="ObjectID" />
```

then the Smarty code needs to loop over each element in that array, testing whether each one matches various conditions. If the first pass declaration instead looks like this:

```
<array name="Titles" table="TMS.ObjTitles" link="ObjectID">
  <filter>{TMS.ObjTitles.Active}=1 and {TMS.ObjTitles.Displayed}=1 and {TMS.ObjTitles.TitleTypeID}=5</filter> <!-- n.b magic number 5 means 'primary'-->
</array>
```

then the second pass can simply output the titles in the array.

So the answers to this question, in no particular order, are:

- it doesn't matter, it's up to you, whatever works
- use the first pass to implement the 'business logic', the second pass to implement presentation.

- 3 Q: I want to change part of the output and I'm not quite clear where I should be doing this

The output for our sample record is using the TMS Object ID instead of the Object Number in the accession number field. What file & where do I change this ?

```
<cdwalite:locationWrap>
  <cdwalite:locationSet>
    ...
    <cdwalite:workID cdwalite:type="accession">140007</cdwalite:workID>
  </cdwalite:locationSet>
</cdwalite:locationWrap>
```

- A: In the first instance, if there's something you don't like in the output, look in second_pass.tpl; which is effectively a specimen CDWA Lite record, decorated with Smarty tags to populate it with data from the object (and some bits of Smarty code that may omit or include sections of CDWA Lite). So this is a good place to start looking. You should search for the nearest bit of CDWA Lite syntax, eg "<cdwalite:locationWrap>", because this will almost certainly appear in second_pass.tpl as literal text. In this case you'll find:

```
<cdwalite:locationSet>
  ...
  <cdwalite:workID cdwalite:type="accession" >{Objects[0]->ObjectID}</cdwalite:workID>
</cdwalite:locationSet>
```

Now you've found the offending part of the template file, you can delete it or edit it. If you want to insert some new dynamic text, then you need to know where that text will come from. In this case, the default first_pass config file loads all the fields retrieved from the TMS Object table into the "Objects" array, so you can replace the expression:

```
{Objects[0]->ObjectID}
```

with

```
{Objects[0]->ObjectNumber}
```

- 4 Q: how do I define OAI-PMH sets?

- A: The tool as currently configured creates sets based on object packages whose names start with the prefix "OAISet", eg "OAISet sample 1". (We played around with various options including making default sets out of departments, but the participants decided that maximum flexibility would be to use TMS object packages only.)

Sets are optional with OAI; you can still query all records, or records within date range, if there are no sets.

Topic

- A2: The path of least resistance to creating sets is to make object packages with suitable names in TMS. The default configuration will retrieve and process these as sets; and this was built as a relatively hard-wired, not especially configurable option.

Strictly however, the processing plug-in just requires two TSV files, one mapping set ids to set names, and one mapping set ids to object ids. So you can generate sets externally any way you like; or for example if you'd like your OAI repository to be automatically divided into sets based on curatorial department, you could do that by generating some additional files, and specifying these in the "SetSrcFiles" attribute in the master script, in place of "TMS/ObjectPackagesOAISets.txt" and "TMS/ObjPkgListOAISets.txt".

- E Detailed programming questions

- 1 Q: Do the joins in the "First Pass" scripts. behave like inner joins or outer joins?

- A: inner joins

For example, consider this array definition in the default first pass script (simplified slightly)

```
<array name="CreatorSet" table="TMS.ConXrefs" link="ID">
  <filter>[TMS.ConXrefs.TableID] = 108</filter> <!-- Magic number means object table -->
  <join table="TMS.Roles" on="TMS.ConXrefs.RoleID" to="RoleID" />
  <filter>[TMS.Roles.RoleTypeID] = 1</filter>
  <join table="TMS.Constituents" on="TMS.ConXrefs.ConstituentID" to="ConstituentID" />
</array>
```

Given an object id of 1001; the array opening tag and first filter element will extract the following relevant records in the TMS.ConXrefs table to create the array:

ConstituentID	TableID	ID	RoleID
2001	108	1001	1
2002	108	1001	2
2003	108	1001	3

This is then joined on the RoleID to the TMS.Roles table, producing this array:

ConstituentID	TableID	ID	RoleID	RoleTypeID	Role
2001	108	1001	1	1	Artist
2002	108	1001	2	2	Donor
2003	108	1001	3	1	After

Which is filtered to this:

ConstituentID	TableID	ID	Role
2001	108	1001	Artist
2003	108	1001	After

And finally the last join to TMS.Constituents produces this:

ConstituentID	TableID	ID	Role	Name
2001	108	1001	Artist	Alan Bufalini
2003	108	1001	After	Charles Davies

(columns simplified in above examples)

- 2 Q: I am wondering if there is any way to do further text processing on fields returned from TMS.

In tracking down object locations we might see something like this "G332; MS" only the first four characters of which we might be interested in displaying.

I have noticed that 'truncate' was not implemented, and strip only works on whitespace. I know I can do this in the SQL Query using a 'substring' call but I am unsure as to where I can pass COBOAT this information (inside the <field /> tags does not make much sense). Also you haven't implemented the regular expressions functions either.

- A: not at present; this has been recorded as a desirable enhancement, but not yet specified

- F Does this look right? How can I debug this nightmare?

- 1 Q: Currently I am at the stage of a full export of our data. I was finally able to get our data into our MySQL database. I am currently looking at 87,499 rows for the metadata_record table, and 0 rows for all the others. Is that correct? Shouldn't there be rows for sets, etc .. ?

- A: The process generates five tables. metadata_record has the CDWA Lite record for each object, and should be populated with as many records as you had objects (less any thrown out by egregious errors, which will have been reported). Two others define sets; if you haven't specified sets (using TMS Object Packages with special names, in the default configuration) these will be empty. This is perfectly legitimate - sets are optional in OAI. The last two tables aren't used at all in this project. OAICatMuseum requires all five tables to exist in the database, but only requires data in the metadata_record table.

- 2 Q: I am looking for sample queries to run against the set. What are the default identifiers, etc?

- A: If you point your browser directly at the OAICatMuseum root directory, you should get a human readable page including links to request forms for the six verbs. The response to the 'Identify' verb should include a sample identifier, which you can specify to GetRecord; if you do ListIdentifiers, specifying the metadata prefix "cdwalite" but leaving everything else blank, you should get a batch of identifiers. In fact you can just do "ListRecords" similarly with just the metadata prefix, and see a batch of records.

- 3 Q: is there an easy way to figure out what's going into arrays?

- A: edit the master script master2Sample.cbcs, changing the 'outputs' parameter to add "data arrays" in addition to "Flat Files", ie
outputs="Flat Files,Data Arrays"

and then run this script ("Generate File Samples") to process your sample set. In addition to flat file CDWA Lite records for each object in the sample set, the output folder ("service/outputfiles" by default) will contain a "DataArrays_XXX.txt" file for each object. This file contains a dump of all the arrays at the end of the first pass for that object. For easy reading, each array is listed twice: once with records horizontal, columns vertical; and once the other way round.

- 4 Q: I keep getting error messages "Unknown Array" with no assistance as to which array it is incorrectly referencing. I have of course pored over the Firstpass, second pass and our cbcs file, but I'm having a very hard time narrowing this particular error generating line down. Any help on how I might get the script to produce more debugging info for me? Something like, a line number, or even a script name "firstpass" or "second_pass"??

- A: unfortunately there is very little such feedback at present - this has been noted as a desirable enhancement.

In regards to this specific cryptic error message. The fact that it doesn't mention an array means that the routine has been passed an empty array name - the 'Unknown Array' message is followed by the name of the unknown array - since you're not seeing anything, the array name is either empty or whitespace. This particular error can only appear in the second pass.