OAI Server Checklist

For the NSDL to successfully harvest your metadata, your OAI Repository **must** be compliant with the OAI-PMH version 2.0 specification, so please read the specification carefully [http://www.openarchives.org/OAI/openarchivesprotocol.html].

Minimally, you will need a basic understanding of:

- **HTTP** - how to send an XML response over HTTP and the basics of GET and POST requests.
- **XML** - the difference between "well-formed" and "valid," and the difference between XML and HTML
- **XML namespaces** - what they are, the difference between the namespace URI and the namespace prefix; what is meant by the "default" namespace; the difference between a namespace URI and a URL.
- **XML schemas** - what they are, how they can be used, how to indicate a particular schema for a particular namespace in your XML, how to find out which schema is being used for a particular namespace in any XML.

We also recommend familiarity with the Guidelines for Repository Implementers [http://www.openarchives.org/OAI/2.0/guidelines-repository.htm].

Checklist of implementation details

Each of the following details needs to be implemented correctly in order to have successful OAI metadata harvesting.

1. responseDate

In all OAI-PMH responses, the responseDate value **must** be the time and date of the OAI server's response in UTC. This **must** be encoded using the "Complete date plus hours, minutes, and seconds" variant of ISO8601: YYYY-MM-DDThh:mm:ssZ. Note the Z at the end, which implies UTC (meaning Greenwich Mean Time, more or less). Example: 2003-10-24T14:05:27Z

2. email

The email address(es) indicated in the Identify response should be valid and messages sent to them should be seen by the appropriate individual(s).

3. granularity

The granularity indicated in the Identify response **must** match the granularity of the datestamp value in served records. That is, if your OAI server supports seconds granularity, then the datestamp value in served records must include seconds (in ISO8601 UTC, of course).

4. OAI identifiers

OAI identifiers uniquely identify an OAI item. Conversely, each OAI item must be uniquely identified by an OAI identifier. An OAI item may have multiple metadata formats and the metadata may be updated from time to time. Updates will change the *datestamp* in the OAI header, but the OAI identifier will remain the same. OAI-PMH says that identifiers must be URIs. We recommend following the Guidelines for OAI Identifiers [http://www.openarchives.org/OAI/2.0/guidelines-oai-identifier.htm] We do not recommend the use of URLs for OAI identifiers.

5. Simple Dublin Core metadata must be supplied for each item as oai_dc

This is required by the OAI-PMH specification. It need not be perfect, particularly if you're also supplying NSDL_DC metadata or Qualified Dublin Core metadata. However, it is prudent to make your oai_dc as accurate as possible and adhere to as many best practices as possible, because Simple Dublin Core is the "lingua franca" of OAI servers, and thus may be used by more harvesters than just the NSDL harvester.

6. ListMetadataFormats

All metadata formats served **must** be indicated, and for each format served, the appropriate XML namespace and XML schema are given, both in the ListMetadataFormats response and in the metadata records themselves. Note that the OAI-PMH element/field/argument "metadataPrefix" is different from the XML prefix. This means

- There is a URL for a schema provided, and the schema URL returns a valid XML schema.
- The XML schema's target namespace is the namespace indicated in ListMetadataFormats.
- The XML schema's target namespace is the same as the namespace of corresponding OAI records' metadata.
- When requesting ListIdentifiers with metadataPrefix=the format, and note one of the returned identifiers, we can then do a GetRecord request for that identifier and format.
- When requesting ListRecords with that metadataPrefix= the format works properly.

7. Every metadata format you serve for every record must cleanly validate its indicated XML schema

The OAI Repository Explorer has a facility to check schema validity of single XML responses [see lower right of the html page at http://oai.dlib.vt.edu/cgi-bin/Explorer/oai2.0/testoai].

8. The "from" and "until" arguments for selective harvesting must correctly return all appropriate metadata records within the range specified

This is required by the OAI specification, and is necessary to allow incremental harvests (of any metadata changed since a particular date). Here's how to test:

- Issue a ListIdentifiers request with metadataPrefix=oai_dc and note one of the returned identifiers.
- Issue a GetRecord request for that identifier and with the oai_dc metadataPrefix, and note the resulting record's datestamp.
- Issue a ListRecords request using "from" and "until" argument values that include the selected record's datestamp. Note that "from" and "until" values are meant to be inclusive.
- NOTE: if your repository supports seconds granularity, then selective harvesting arguments can have seconds granularity.

9. resumptionTokens

We recommend using resumptionTokens if you need to serve more than 2 megabytes of metadata. If you use resumptionTokens, then:

- they must be implemented properly per the OAI-PMH specification.
    - They must appear correctly in affected responses, in the correct location.
    - When present, they must work; a request using the resumptionToken should return appropriate results.
    - The last response of a complete list must have an empty resumption token.
- The chunk size should be reasonable. Ideal response size is probably between 1 and 2 Meg.
- resumptionTokens in the response should NOT be URL encoded; resumptionTokens in a request MUST be URL encoded.
- Recommendation: do not use characters in your resumption tokens that require URL encoding.

10. ListRecords harvesting works

We can harvest all your metadata using ListRecords. That is, ListRecords works, and resumptionTokens work properly with ListRecords, and we get the correct metadata format when we request it.

11. sets

If you've implemented sets:

- ListSets properly indicates all sets served and gives us the appropriate information about the sets, per the OAI-PMH specification.
- Set membership is correctly indicated with a <setSpec> element in the OAI header of the affected records.
- Selective harvesting by set works properly -- a ListRecords request with a set argument only retrieves records in the specified set