Interoperability – Digital Library Protocols

CS 431 – March 13, 2006
Carl Lagoze – Cornell University

Dienst Problem Space

- Early web (1994)
- CS scholars putting pre-prints (tech reports) on web pages
  - Non-persistence (life of the research project)
  - No federated searching
- Institutional identity and control important
- Exploit WWW technology to create federated digital library

Dienst

- is a protocol and reference implementation of a distributed digital library service
- where a network of services provide
- World Wide Web browser access,
- uniform search over distributed indexes,
- and access to structured documents.

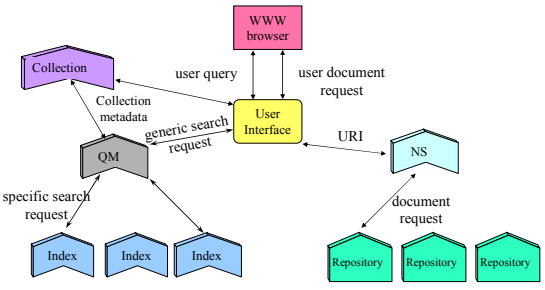
Why a service based protocol?

- Expose the operational semantics of the services through an API,
- to permit flexible integration of the services,
- and use of the services by other clients/consumers/services.

Defining the services

- Repository – deposit, storage, and access to structured documents.
- Index – process queries on documents and returned handles
- Query Mediator – route queries to appropriate indexes
- Collection – define services and content in logical collections
- User Interface – human-oriented front-end for services.
- Name Server – Resolves URN's (handles) to document location(s)

Dienst Services



Defining the protocol

- Structured messages
  - Service
  - Version
  - Verb
  - Arguments
- Template
  /Dienst/<service>/<version>/<verb>[?/]<arguments>
- Example
  /Dienst/Repository/4.0/Formats/ncstrl.cornell/TR94-1418

Why a Document Model?

- Documents are multi-faceted
  - Formats
  - Decomposition
  - Metadata vs. data

Dienst Document Model

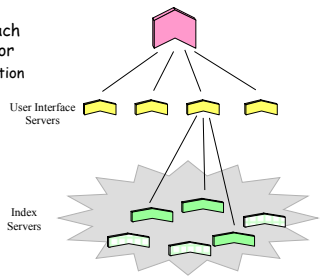
- *Metadata* – support for multiple descriptive formats
- *Views* – alternative expression or structural representation of the content encapsulated in the digital object
- *Divs* – hierarchically nested structure contained in a view

Expressing the document model in the protocol

- *Structure* – expose the views and structure for the digital object
- *Disseminate* – select the structural component (and packaging of it) to disseminate
- *List-Meta-Formats* – list available descriptive formats

Collection Service

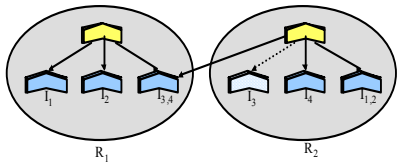
- Defines the scope of the library by what can be searched
- Periodically polled by each user interface server for
  - elements of the collection
  - index servers for the collection



Deploying Collection Globally

- Internet connectivity varies considerably
- Good connectivity between nodes often does not correspond to geographic proximity
- *Connectivity Region* - a group of nodes on the network that among them have good connectivity, relative to nodes outside of the region.

Connectivity Regions



- When possible keep library activity within region
- In case of failure, use an alternate either within the region or in a "nearby" region

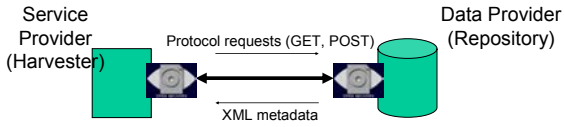
Open Archives Initiative Problem Space

- Limited success of Dienst concept
  - NCSTRL peak 120 sites
- Document level interoperability not generally accepted
- Increase in heterogeneous pre-print servers and institutional repositories
  - Same federation problem
- Increase in crawlers
  - Need to solve deep web problem for repositories
- Maturation of foundation standards
  - XML
  - Dublin Core

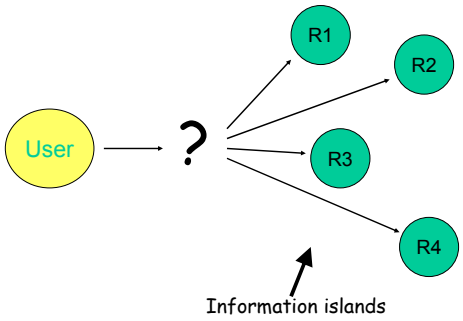
OAI-PMH

⇒ PMH -> Protocol for Metadata Harvesting
http://www.openarchives.org/OAI/2.0/openarchivesprotocol.htm

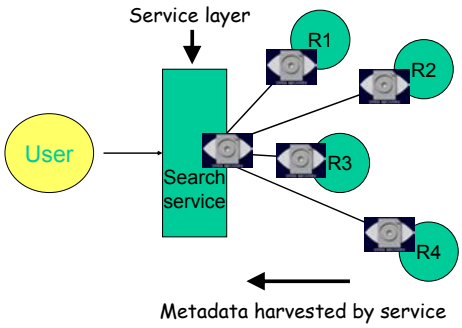
- Simple protocol, just 6 verbs
- Designed to allow harvesting of any XML (meta)data (schema described)
- For batch-mode not interactive use



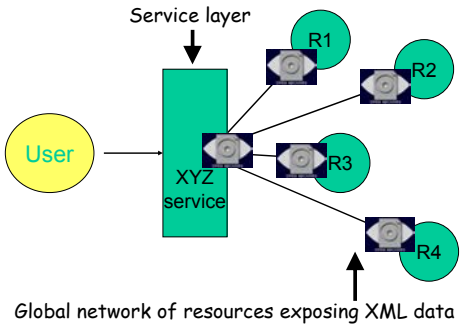
OAI for discovery



OAI for discovery



OAI for XYZ



OAI-PMH Data Model

resource

item has identifier

all available metadata about this sculpture

item

Dublin Core metadata | MARC21 metadata | branding metadata

records

record has identifier + metadata format + datestamp

OAI and Metadata Formats

- Protocol based on the notion that a record can be described in multiple metadata formats
- Dublin Core is required for "interoperability"
- Extended to include XML compound object formats: e.g., METS, DIDL
  - http://www.dlib.org/dlib/december04/vandesompel/12vandesompel.html

OAI-PMH and HTTP

- OAI-PMH uses HTTP as transport
  - Encoding OAI-PMH in GET
    - http://baseURL?verb=<verb>&arg1=<arg1Val>...
    - Example: http://an.oa.org/OAIscript?verb=GetRecord&identifier=oai:arXiv.org:hep-th/9901001&metadataPrefix=oai_dc
- Error handling
  - all OK at HTTP level? => 200 OK
  - something wrong at OAI-PMH level? => OAI-PMH error (e.g. badVerb)
- HTTP codes 302 (redirect), 503 (retry-after), etc. still available to implementers, but do not represent OAI-PMH events

OAI-PMH verbs

	Verb	Function
metadata about the repository	Identify	description of archive
	ListMetadataFormats	metadata formats supported by archive
	ListSets	sets defined by archive
harvesting verbs	ListIdentifiers	OAI unique ids contained in archive
	ListRecords	listing of N records
	GetRecord	listing of a single record

most verbs take arguments: dates, sets, ids, metadata formats and resumption token (for flow control)

Identify verb

Information about the repository, start any harvest with Identify

```
<Identify>
    <repositoryName>Library of Congress 1</repositoryName>
    <baseURL>http://memory.loc.gov/cgi-bin/oai</baseURL>
    <protocolVersion>2.0</protocolVersion>
    <adminEmail>r.e.gillian@larc.nasa.gov</adminEmail>
    <adminEmail>rgillian@visi.net</adminEmail>
    <deletedRecord>transient</deletedRecord>
    <earliestDatestamp>1990-02-01T00:00:00Z</earliestDatestamp>
    <granularity>YYYY-MM-DDThh:mm:ssZ</granularity>
    <compression>deflate</compression>
```

GetRecord - Normal response

```
<?xml version="1.0" encoding="UTF-8"?>
<OAI-PMH>        ....namespace info not shown here
<responseDate>2002-0208T08:55:46Z</responseDate>
<request verb="GetRecord"… …>http://arXiv.org/oai2</request>
 <GetRecord>
  <record>
   <header>
    <identifier>oai:arXiv:cs/0112017</identifier>
    <datestamp>2001-12-14</datestamp>
    <setSpec>cs</setSpec>
    <setSpec>math</setSpec>
   </header>
   <metadata>
    …..
   </metadata>
  </record>
 </GetRecord>
</OAI-PMH>
```

Error/exception response

```
<?xml version="1.0" encoding="UTF-8"?>
<OAI-PMH>
<responseDate>2002-0208T08:55:46Z</responseDate>
<request>http://arXiv.org/oai2</request>
<error code="badVerb">ShowMe is not a valid OAI-PMH verb</error>
</OAI-PMH>
```

Same schema for all responses, including error responses.

with errors, only the correct attributes are echoed in <request>

Identifiers

- Record is self-contained object that may stored/manipulated etc.
- Items have identifiers (all records of same item share identifier)
- Complete identification of a record is baseURL+identifier+metadataPrefix+datestamp

```
oai-identifier = scheme ":" namespace-identifier ":" local-identifier

scheme = "oai"

namespace-identifier = domainname-word "." domainname
domainname = domainname-word [ "." domainname ]
domainname-word = alpha *( alphanum | "-" )
```

oai:arXiv.org:hep-th/9901001

oai:foo.org:some-local-id-53

Selective Harvesting

- RSS is mainly a "tail" format
- OAI-PMH is more "grep" like
- Two "selectors" for harvesting
  - Date
  - Set
- Why not general search?
  - Out of scope
  - Not low-barrier
  - Difficulty in achieving consensus

Datestamps

- All dates/times are UTC, encoded in ISO8601, Z notation: `1957-03-20T20:30:00Z`
- Datestamps may be either fill date/time as above or date only (YYYY-MM-DD). Must be consistent over whole repository, 'granularity' specified in Identify response.
- Earlier version of the protocol specified "local time" which caused lots of misunderstandings. Not good for global interoperability!

Harvesting granularity

- mandatory support of `YYYY-MM-DD`
- optional support of `YYYY-MM-DDThh:mm:ssZ` (must look at Identify response)
- granularity of `from` and `until` agrument in ListIdentifier/ListRecords must match

Sets

- Simple notion of grouping at the item level to support selective harvesting
  - Hierarchical set structure
  - Multiple set membership permitted
  - E.g: repo has sets A, A:B, A:B:C, D, D:E, D:F
    If item1 is in A:B then it is in A
    If item2 is in D:E then it is in D, may also be in D:F
    Item3 may be in no sets at all

Record headers

- `header` contains set membership of item

```
<record>
  <header>
   <identifier>oai:arXiv:cs/0112017</identifier>
   <datestamp>2001-12-14</datestamp>
   <setSpec>cs</setSpec>
   <setSpec>math</setSpec>
  </header>
  <metadata>
   …..
  </metadata>
 </record>
```

resumptionToken

- Protocol supports the notion of partial responses in a very simple way: Response includes a 'token' at the which is used to get the next chunk.
- Idempotency of `resumptionToken`: return same incomplete list when `resumptionToken` is reissued
  - while no changes occur in the repo: strict
  - while changes occur in the repo: all items with unchanged `datestamp`
  - optional attributes for the resumptionToken: `expirationDate`, `completeListSize`, `cursor`

Harvesting strategy

- Issue Identify request
  - Check all as expected (validate, version, baseURL, granularity, comporession…)
- Check sets/metadata formats as necessary (ListSets, ListMetadataFormats)
- Do harvest, initial complete harvest done with no `from` and `to` parameters
- Subsequent incremental harvests start from datastamp that is responseDate of last response

OAI-PMH – Has it worked?

- Of course, yes…
  - Very wide deployment
  - "millions and millions of records served"
  - Incorporated into commercial systems
- But….
  - NSDL experience has shown "low barrier" is not always true
    - XML is hard
  - Incremental harvesting model is full of holes