Fedora and GSearch in a Research Project about Integrated Search

Open Repositories 2009

Gert Schmeltz Pedersen
DTU Library, Technical Information Center
@ DTU, Technical University of Denmark
Overview

- Introduction
- Isearch experimental setup
- GSearch indexing and configuration
- Fedora+GSearch for Isearch experimental setup
- Details of harvest, ingest and indexing
- Integrated search
- Conclusion
Introduction

➲ The Royal School of Library and Information Science[1] in Denmark is performing a research project about integrated search, partly funded by DEFF[2], Denmark's Electronic Research Library. DTU Library provides assistance in the form of a Fedora+GSearch installation.

➲ The research project involves empirical investigations following the principles of polyrepresentation in information retrieval[3]:

- “The polyrepresentation principle suggests that cognitively and functionally different representations of information objects may be used in information retrieval to enhance quality of results. …. The continuum proposed by Larsen (2004; Ingwersen & Larsen, 2005) showing the structural dimension of the retrieval techniques involved in polyrepresentation is further elaborated by adding a novel second dimension consisting of query structure and modus. The new two-dimensional continuum can be seen as a constructive framework for further investigations of polyrepresentative principles in IR.”

The Fedora+GSearch installation provides a backend for the empirical investigations, containing an index of metadata records harvested from various sources, thus allowing polyrepresentation of information objects.

The installation so far harvests from two types of sources, one using OAI-PMH[4], and one using the SRU protocol[5].

This presentation will focus on the technical challenges involved in the setup and indexing of the various sources, facilitating the integrated search.

Isearch experimental setup

- **User information needs**, ~50, formulated by *domain experts* in physics, translated to queries by the *Isearch researchers*
- Sets of **metadata** (and possibly **full-texts**) characterized by origin, format, etc.
- **Indexes** built on subsets of metadata, characterized by origin and index process properties
- **Search results** for user information needs
- **Evaluations** of search results vs user information needs by *domain experts*
- **Isearch webapp**
  - Receives metadata and full-texts
  - Performs searches
  - Provides search sets for evaluation
  - Stores evaluations for analysis by researchers
  - Allows various **configurations** of metadata and indexes, for experimentation
- The harvest and ingest service does
  - **Harvest** by various protocols, data formats, and data providers
  - **Ingest** according to configurations
- Thus, we may have **polyrepresentation**, in that a publication may have many different representations
- Each configuration will lead to a different set of search results for the set of user information needs, so that the *Isearch researchers* can use the domain expert evaluations to **conclude** on effects of polyrepresentation
- **Fedora with GSearch** was selected to assist the tasks in green. Some of the considerations follow
GSearch indexing

- Clients interact with the fedora webapp for management of fedora objects (foxml records) and access to them.
- Clients interact with the gsearch webapp for indexing, search and browse of foxml records.
- Fedora may be configured to send indexing messages to gsearch, when foxml records are managed.
- The gsearch updateIndex operation (green) consists of a transformation of a foxml record into an IndexDocument, which is delivered to the search engine for inclusion in the index.
- The transformation is given in an xslt stylesheet, where each IndexField gets its contents from somewhere in the foxml record, either from object properties, or from inline xml datastreams, or from managed, external or referenced datastreams, which could be full-texts.
GSearch - many-to-many

- GSearch may be configured to receive foxml records from many fedora repositories and to maintain many indexes, in parallel.

- Foxml records may be indexed by many different stylesheets, and thus give different IndexDocuments for the different indexes.
**GSearch configuration**

**Configuration example:**

- fedoragsearch.properties
  - soapBase = http://HOSTPORT/fedoragsearch/services
  - repositoryNames = REPOSNAMES
  - indexNames = INDEXNAMES

- INDEXNAME/index.properties
  - operationsImpl = dk.defxws.fgslucene.OperationsImpl
  - defaultQueryFields = dc.description dc.title

- REPOSNAME/repository.properties
  - soapBase = http://FEDORAHOSTPORT/fedora/services
  - fedoraObjectDir = FEDORAOBJECTDIR
Fedora+GSearch for Isearch experimental setup

- The Isearch webapp is realized by the two webapps and the gsearch config
- We use the lucene search engine
- The harvest service is so far realized for two protocols, OAI-PMH and SRU
- The ingest service is realized by format specific xslt stylesheets and some gluing code. These stylesheets basically wraps the harvested records as datastreams into foxml records
- The indexing xslt stylesheets are tailored to fetch the contents for the IndexFields from the contents of specific elements in the metadata records (examples follow)
- Many different gsearch configs may be provided, each will determine a subset of metadata records and characteristics of the resulting indexes, and will thereby lead to a different set of search results for the set of user information needs (run as gfindObjects queries from a client)
**Harvest and ingest with OAI-PMH**

- OAI-PMH, Open Archives Initiative Protocol for Metadata Harvesting, is a mechanism for repository interoperability

- The harvester2 library from OCLC has been downloaded and installed, run example:

  ```
  java  
  -cp :log4j-1.2.12.jar:harvester2.jar:xalan.jar
  ORG.oclc.oai.harvester2.app.RawWrite
  -from 2009-01-01
  -until 2009-04-30
  -setSpec physics
  -metadataPrefix oai_dc
  http://export.arxiv.org/oai2
  ```
Harvest with OAI-PMH

-- ListRecords --

function
harvest records from a repository

example
archive.org/oai-script?verb=ListRecords&
metadataPrefix=oai_dc&set=biology

parameters
from (optional)
until (optional)
metadataPrefix (required)
set (optional)
resumptionToken (exclusive)

errors / exceptions
badArgument
badResumptionToken
cannotDisseminateFormat
noRecordsMatch
noSetHierarchy

-- GetRecord --

function
retrieve individual metadata record from a repository

example
archive.org/oai-script?verb=GetRecord&
identifier=oai:HU-Berlin:de:3000218&
metadataPrefix=oai_dc
OAI-PMH record and indexing example

OAI-PMH record and indexing example

<IndexDocument>
  <IndexField IFname="TITLE" index="TOKENIZED" store="YES" termVector="NO" boost="1.0">
    Using Structural Metadata ...
  </IndexField>
  <IndexField IFname="CREATOR" index="TOKENIZED" store="YES" termVector="NO" boost="1.0">
    Dushay, Naomi
  </IndexField>
  <IndexField IFname="SUBJECT" index="TOKENIZED" store="YES" termVector="NO" boost="1.0">
    Digital Libraries
  </IndexField>
</IndexDocument>
Harvest and indexing with SRU

➲ SRU – Search/Retrieval via URL

➲ Example:

http://webservice.bibliotek.dk/soeg/?version=1.1

&operation=searchRetrieve
&query=dc.title=physics
&startRecord=1
&maximumRecords=100
&recordSchema=marcX
&stylesheet=default.xsl
&recordPacking=xml
Harvest with SRU

The Library of Congress > Standards > SRU

SRU Search/Retrieval via URL

SRU (the protocol)
CQL (query language)
ZeeRex (service description)

sruthome > sruversion 1.2 specs > searchretrieve operation

searchRetrieve Operation (SRU Version 1.2 Specifications)

SECTIONS: Parameters | Records | Result Sets

The searchRetrieve operation is the main operation in SRU. It allows the client to submit a search and retrieve request for matching records from the server.

PARAMETERS

Request Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Mandatory/Optional?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>operation</td>
<td>mandatory</td>
<td>The string: 'searchRetrieve'.</td>
</tr>
<tr>
<td>version</td>
<td>mandatory</td>
<td>The version of the request, and a statement by the client that it wants the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>response to be less than, or preferably equal to, that version. See Version.</td>
</tr>
<tr>
<td>query</td>
<td>mandatory</td>
<td>Contains a query expressed in CQL to be processed by the server. See CQL.</td>
</tr>
<tr>
<td>startRecord</td>
<td>optional</td>
<td>The position within the sequence of matched records of the first record to</td>
</tr>
<tr>
<td></td>
<td></td>
<td>be returned. The first position in the sequence is 1. The value supplied</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MUST be greater than 0. The default value if not supplied is 1.</td>
</tr>
<tr>
<td>maximumRecords</td>
<td>optional</td>
<td>The number of records requested to be returned. The value must be 0 or</td>
</tr>
<tr>
<td></td>
<td></td>
<td>greater. Default value if not supplied is determined by the server. The</td>
</tr>
<tr>
<td></td>
<td></td>
<td>server MAY return less than the number of records, for example if there are</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fewer matching records than requested, but MUST NOT return more than this</td>
</tr>
<tr>
<td></td>
<td></td>
<td>number of records.</td>
</tr>
<tr>
<td>recordPacking</td>
<td>optional</td>
<td>A string to determine how the record should be escaped in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Defined values are 'string' and 'xml'. The default is 'xml'. See Records.</td>
</tr>
<tr>
<td>recordSchema</td>
<td>optional</td>
<td>The schema in which the records MUST be returned. The value is the URI</td>
</tr>
<tr>
<td></td>
<td></td>
<td>identifier for the schema or the short name for it published by the server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default value if not supplied is determined by the server. See Records.</td>
</tr>
<tr>
<td>recordXPath</td>
<td>optional</td>
<td>An XPath expression, to be applied to the records before returning them. It</td>
</tr>
</tbody>
</table>
|                  |                     | is to be
Harvest with SRU

Webservice til søgning i bibliotek.dk

bibliotek.dk SRU/GET og SRU/SOAP webservice

dc.title (Titel) = physics

and

dc.date (År) = 2008

dc.title = (physics) and dc.date = (2008)

startRecord: 1
maximumRecords: 10
recordSchema: marcXchange

searchRetrieval

CQL operators og relations

proximity
booleanModifier
booleanModifier
booleanModifier
booleanModifier
relation
relation
relation
relation
relation
maskingCharacter

Særligt for dc.type og dc.language

Der anvendes følgende: materialekoder og sprogkoder
SRU record and indexing example

```xml
<xml version="1.0" encoding="UTF-8"?>
<record xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.openarchives.org/sru/1.1
                          http://www.openarchives.org/sru/1.1/schema"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:mxc="http://www.loc.gov/MARC21/xslt"
        xmlns:foxml="http://www.openarchives.org/foxml/

<IndexDocument>
  <IndexField IFname="CREATOR" index="TOKENIZED" store="YES" termVector="NO" boost="1.0">
    <xsl:value-of select="text()"/>
  </IndexField>
  <IndexField IFname="TITLE" index="TOKENIZED" store="YES" termVector="NO" boost="1.0">
    <xsl:value-of select="text()"/>
  </IndexField>
  <IndexField IFname="SUBJECT" index="TOKENIZED" store="YES" termVector="NO" boost="1.0">
    <xsl:value-of select="text()"/>
  </IndexField>
</IndexDocument>
```
Integrated search

➲ Now we may search on all sources in one query

➲ Examples:
  ● CREATOR:Smith and SUBJECT:oxygen
  ● Smith and oxygen and TITLE:process*

➲ Queries may be on
  ● specific fields
  ● and/or on the configured defaultQueryFields

➲ Search results may be from all sources indexed
  ● in the default index
  ● or in a specific index named in the query URL
Conclusion

- The delivery to Isearch is an example of a non-trivial and unusual application of Fedora+GSearch
- The facilities for experimentation are key points for Isearch
- The delivery has yet to prove its value
- Hopefully, you have been inspired to see new possibilities for your own projects with Fedora+GSearch