Review of the Tools and Software to Support Interoperability (Work Area B.1)

Serhiy Polyakov

William E. Moen
wemoen@unt.edu

Mark Phillips
MPhillips@library.unt.edu

July 15, 2006
Draft 1
<table>
<thead>
<tr>
<th>Version</th>
<th>Date and Time of Modification</th>
<th>Name of Modifier</th>
<th>Section Modified</th>
<th>Brief Description of the Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Draft 1</td>
<td>July 1 2006</td>
<td>Polyakov</td>
<td>All</td>
<td></td>
</tr>
</tbody>
</table>
Table of Contents

1. Introduction ................................................................................................................................. 1
2. Tools/Software .............................................................................................................................. 2
   2.1. SimpleServer ......................................................................................................................... 2
   2.2. Zebra ..................................................................................................................................... 3
   2.3. Lucene / SRWLucene........................................................................................................... 5
   2.4. ZContent: ............................................................................................................................ 7
   2.5. Cheshire ............................................................................................................................... 8
3. OAI-PMH tools ............................................................................................................................ 9
4. Summary ....................................................................................................................................... 11
1. Introduction

This document provides an overview of tools and software to improve interoperability between digital repositories and the Library of Texas federated search. This review will be used for providing recommendations for best practices and workflows for installing software and tools onto Digital Asset Management Systems (DAMS) and selected databases that support search interoperability.

This document examines the following tools/software:
- SimpeServer (Index Data)
- Zebra (Index Data)
- Lucene with SRU/SRW support (University of Oregon)
- ZContent (University of Utah)
- Cheshire3 (UC Berkeley)

Figure 1 illustrates the connections between Digital Repositories and THDI web portal.

Figure 1. Connection between Digital Repositories and THDI web portal

(*) "...installing Z39.50 and/or SRU/SRW gateways to both closed and open databases ... create and maintain static OAI repositories..."

(**) "...one of the results of THDI's IMLS grant will be an OAI harvester developed and run by the University of North Texas, which can then be searched via Z39.50 from the main THDI search interface..."
2. Tools/Software

2.1. SimpleServer

Possible applications in THDI project
Z39.50, SRU or SRW Gateways to Digital Repositories.

Developer
Index Data <http://www.indexdata.dk/>

Overview
Net::Z3950::SimpleServer is a Perl module (API) which is intended to develop new Z39.50, SRU and SRW servers over any type of database imaginable.

The programmer has to implement functions handling:
- Initializing of a database (optional),
- searching the database,
- returning "database records" on request,
- sorting (optional),
- scanning indexes (optional),
- connection closure (optional), and
- eventually other Z39.50/SRU/SRW events such as result-set deletion and extended services.

The module takes care of everything else and automatically starts a server, listens to incoming connections, and implements the Z39.50, SRU and SRW protocols.

It is robust, efficient, widely portable, and it inter-operates with all known Z39.50, SRU and SRW clients.

Other toolkits/plugging requirements
YAZ toolkit

Usage/Programming Language
Use SimpleServer together with other Perl modules to provide gateways to relational databases, local file stores, SOAP/RDF-servers, etc.

Functionality/Services
Implements a "generic" Z39.50 server, which invokes callback functions supplied by the programmer to search for content in a database. The programmer can use any tools available in Perl to supply the content, including modules like DBI and WWW::Search.

SimpleServer currently supports the Init, Search, Present, Sort, Scan and Close services.

DBMS Supported
Any if appropriate functions are developed

Documentation
Limited

Installation and administration complexity
Moderate. Knowledge of OO programming in Perl is required.

OS Platforms
Linux, Windows

Versions
Version 1.01, Released 2006-03-24

**Size of the installation package**
~100KB

**Support**
The SimpleServer module can be covered by the optional Index Data support agreement.

**Sources used**


### 2.2. Zebra

**Possible applications in THDI project**
?

**Developer**
Index Data [http://www.indexdata.dk/](http://www.indexdata.dk/)

**Overview**
Zebra is a high-performance, general-purpose structured text indexing and retrieval engine. It reads structured records in a variety of input formats (eg. email, XML, MARC) and allows access to them through exact boolean search expressions and relevance-ranked free-text queries.

Zebra supports large databases (more than ten gigabytes of data, tens of millions of records). It supports incremental, safe database updates on live systems. Data stored in Zebra can be accessed using a variety of Index Data tools (eg. YAZ and PHP/YAZ) as well as commercial and freeware Z39.50 clients and toolkits.

**Licence**
Zebra is free software, available under the GPL license. It may be used by anyone without charge.

**Versions**
Version 1.3.36, Released 2006-06-07

**Size of the installation package**
~2MB

**OS Platforms**
Unix-Like, Windows

**Features:**
- Very large databases: logical files can be automatically partitioned over multiple disks.
- Arbitrarily complex records. The internal data format is a structured format conceptually similar to XML or GRS-1, which allows lists, nested structured data elements and variant forms of data.
- Robust updating - records can be added and deleted “on the fly” without rebuilding the index from scratch. Records can be safely updated even while users are accessing the server. The update procedure is tolerant to crashes or hard interrupts during database updating - data can be reconstructed following a crash.
• Configurable to understand many input formats. A system of input filters driven by regular expressions allows most ASCII-based data formats to be easily processed. SGML, XML, ISO2709 (MARC), and raw text are also supported.
• Searching supports a powerful combination of boolean queries as well as relevance-ranking (free-text) queries. Truncation, masking, full regular expression matching and "approximate matching" (eg. spelling mistakes) are all handled.
• Index-only databases: data can be, and usually is, imported into Zebra’s own storage, but Zebra can also refer to external files, building and maintaining indexes of "live" collections.

Z 39.50 Protocol Support
Z39.50 protocol facilities: Init, Search, Present (retrieval), Segmentation (support for very large records), Delete, Scan (index browsing), Sort, Close and support for the “update” Extended Service to add or replace an existing XML record

Documentation
Extensive. Mailing list.

Support
Web Site, Mailing List, Commercial support available.

Installation and administration complexity
Simple to Moderate

Exchange Formats
Converting records from the internal structure to an exchange format is largely an automatic process.

Currently, the following exchange formats are supported:
• GRS-1. The internal representation is based on GRS-1/XML, so the conversion here is straightforward. The system will create applied variant and supported variant lists as required, if a record contains variant information.
• XML. The internal representation is based on GRS-1/XML so the mapping is trivial. Note that XML schemas, preprocessing instructions and comments are not part of the internal representation and therefore will never be part of a generated XML record. Future versions of the Zebra will support that.
• SUTRS. Again, the mapping is fairly straightforward. Indentation is used to show the hierarchical structure of the record. All "GRS" type records support both the GRS-1 and SUTRS representations.
• ISO2709-based formats (USMARC, etc.). Only records with a two-level structure (corresponding to fields and subfields) can be directly mapped to ISO2709. For records with a different structuring (eg., GILS), the representation in a structure like USMARC involves a schema-mapping to an "implied" USMARC schema (implied, because there is no formal schema which specifies the use of the USMARC fields outside of ISO2709). The resultant, two-level record is then mapped directly from the internal representation to ISO2709.
• Explain. This representation is only available for records belonging to the Explain schema.
• Summary. This ASN-1 based structure is only available for records belonging to the Summary schema - or schema which provide a mapping to this schema (see the description of the schema mapping facility above).
• SOIF. Support for this syntax is experimental, and is currently keyed to a private Index Data OID (1.2.840.10003.5.1000.81.2). All abstract syntaxes can be mapped to the SOIF format, although nested elements are represented by concatenation of the tag names at each level.

Query Language
For searching, Zebra accepts a query and a set of databases to be searched. Natively Zebra supports only RPN queries (the Type-1 query from Z39.50), but it comes with tools that make it possible to use CQL (http://www.loc.gov/z3950/agency/zing/cql/) or CCL (ISO8777), given a mapping file to RPN. For
RPN, Zebra supports a string notation, PQF, that is used by YAZ. It expresses an RPN query in a string format--unlike Z39.50 RPN queries, which are expressed in ASN.1.

**Sources used**


**2.3. Lucene / SRWLucene**

**Possible applications in THDI project**

**Developer**
Original Developer is Doug Cutting. Lucene is maintained by a team of volunteer developers. Apache Lucene is an open source project available for free download.

**Overview**
Apache Lucene is a high-performance, full-featured text search engine library written entirely in Java. It is a technology suitable for nearly any application that requires full-text search, especially cross-platform.

Lucene allows adding indexing and searching capabilities to the applications. Lucene can index and make searchable any data that can be converted to a textual format. Similarly, with Lucene it is possible to index data stored in the databases, providing full-text search capabilities that many databases don't provide.

Although Lucene only supports simple text, there are Java classes that are available that can convert HTML, XML, Word documents, and PDF files into simple text.

...From Lucene FAQ...
To index a database, connect to the database using JDBC and use an SQL "SELECT" statement to query the database. Then create one Lucene Document object per row and add it to the index. You will probably want to store the ID column so you can later access the matching items. For other (text) columns it might make more sense to only index (not store) them, as the original data is still available in your database.

**Versions**
26 May 2006 - Release 2.0.0 available

**Features**
Lucene offers powerful features through a simple API.

**Scalable, High-Performance Indexing**
- over 20MB/minute on Pentium M 1.5GHz
- small RAM requirements -- only 1MB heap
- incremental indexing as fast as batch indexing
- index size roughly 20-30% the size of text indexed

**Powerful, Accurate and Efficient Search Algorithms**
- ranked searching -- best results returned first
Review of the Tools and Software to Support Interoperability

- many powerful query types: phrase queries, wildcard queries, proximity queries, range queries and more
- fielded searching (e.g., title, author, contents)
- date-range searching
- sorting by any field
- multiple-index searching with merged results
- allows simultaneous update and searching

Cross-Platform Solution
- Available as Open Source software under the Apache License which lets you use Lucene in both commercial and Open Source programs
- 100%-pure Java
- implementations in other programming languages available that are index-compatible

Installation and administration complexity
Moderate. Thorough knowledge of OO programming in Java is required.

Documentation
Extensive (Book, forum, FAQ, online)

SRWLucene
SRWLucene is an addon to the SRW/U 2.0 from OCLC. This add-on is designed to allow SRW/U support for any database with a Lucene index.

Lucene is only an indexing engine. It is used in conjunction with some storage mechanism that contains the actual records. SRWLucene defines a RecordResolver Interface that provides the link between itself and a specific storage mechanism.

RecordResolver provides the following functionality:

- Resolves records from identifiers
- Determines which schemas are supported
- Generates SchemaInfo xml for use in explain responses
- Transforms records from their default schema to another

Sources used


2.4. ZContent:

Possible applications in THDI project
Z39.50 Gateway to Digital Repositories based on CONTENTdm digital collections server.

Developer
The University of Utah Marriott Library

Overview
ZContent is an Open Source Z39.50 Connection to CONTENTdmTM Digital Collection Management Software. It adds Z39.50 compatibility to any CONTENTdm digital collections server. The purpose of this software is to provide access to digital collections on CONTENTdm servers from library portals and local catalogs. ZContent will process incoming Z39.50 requests, convert them to appropriate CONTENTdm requests, and return the results in either USMARC or XML format.

Functionality
ZContent is a Perl script and module that provides a Z39.50 target for the CONTENTdm server. Perl versions 5.6.1 and 5.8 will both work.

The communication takes place as follows:
Z39.50 Client \(=>\) z39.50 \(==>\) ZContent \(==>\) HTTP \(==>\) CONTENTdm

ZContent is based on the open source SimpleServer Perl module which is provided by Indexdata. SimpleServer is based on the YAZ toolkit, which is also provided by Indexdata. USMARC Records are created using the MARC-Record perl module.

Licence
ZContent is available under a GNU General Public License (GPL).

Installation and administration complexity
Moderate.

Size of the installation package
<100KB

Documentation
Limited (readme file)

Dependencies
YAZ and SimpleServer has to be installed first.

Sources used
http://www.lib.utah.edu/digital/about/zcontent.php

2.5. Cheshire

Possible applications in THDI project

Developer
Ray R. Larson
University of California, Berkeley

Overview
Cheshire3 Information Retrieval Framework is a fast XML search engine, written in Python for extensibility and using C libraries for speed. It is feature rich, including support for XML namespaces, unicode, a distributable object oriented model and all the features expected of a digital library system. Standards are foremost, including SRW/U and CQL, as well as Z39.50 and OAI. It is highly modular and configurable, enabling very specific needs to be addressed with a minimum of effort. The API is stable and fully documented, allowing easy third party development of components.

Functionality
Given a set of records, Cheshire3 can extract data into one or more indexes after processing with configurable workflows to add extra normalisation and processing. Once the indexes have been constructed, it supports such operations as search, retrieve, browse and sort. Using Apache handlers, any interface from a shop front, to Z39.50 to OAI can be provided (all included by default), but the abstract protocolHandler allows integration into any environment that will support Python.

There are two main processes that Cheshire3 takes care of:

1. The preprocessing 'ingestion' phase in which data is taken into the system and processed. This phase takes into account such things as cleaning the data, extracting the terms from the records and storage of the data and terms.
2. The real time 'discovery' phase in which a user interacts with the system to find items that match their information need.

Installation and administration complexity
Moderate to Complex. Knowledge of OO programming in Python is required for interface development.

Other toolkits/plugging requirements
BerkeleyDb, Python, 4Suite, ZSI (Python SOAP toolkit), PyZ3950,

Versions
Version 3

Size of the installation package
~100KB

Documentation
Some documentation available online

Sources used

## 3. OAI-PMH tools

This is the list of OAI-PMH tools implemented by members of the Open Archives Initiative community.  

<table>
<thead>
<tr>
<th>Tool</th>
<th>Implementer</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arc source</td>
<td>Old Dominion University</td>
<td>Arc is released under the NCSA Open Source License. Arc is a federated search service based on OAI-PMH. It includes a harvester, a search engine together with a simple search interface, and an OAI-PMH layer over harvested metadata. Arc can be configured for a specific community, and enhancements and customizations by the community are encouraged. Arc is based on Java Servlet technology and requires JDK1.4, Tomcat 4.0x, and a RDBMS server (tested with Oracle and MySQL).</td>
</tr>
<tr>
<td>Archimede</td>
<td>Laval University Library CERN</td>
<td>Archimede is an open-source software for institutional repositories. It features full text searching, multiplatform support, Web user interface, and more. Archimede fully supports OAI-PMH requests version 2.0.</td>
</tr>
<tr>
<td>CDSware</td>
<td>CERN</td>
<td>The CERN Document Server Software (CDSware) allows you to run your own electronic preprint server. It complies with OAI-PMH and uses MARC 21 as its underlying bibliographic standard.</td>
</tr>
<tr>
<td>Celestial 1.1.1</td>
<td>University of Southampton</td>
<td>Celestial is an OAI 2.0-compliant OAI aggregation/caching/mirroring tool. Using MySQL and Perl, Celestial allows the administrator to mirror 1.0, 1.1, and 2.0 repositories in their own 2.0-compliant repository.</td>
</tr>
<tr>
<td>DP9</td>
<td>Old Dominion University</td>
<td>An OAI Gateway Service for Web Crawlers</td>
</tr>
<tr>
<td>DSpace</td>
<td>HP Labs and MIT Libraries</td>
<td>DSpace is an open source digital asset management software platform that enables institutions to capture and describe digital content. It runs on a variety of hardware platforms and supports OAI-PMH version 2.0.</td>
</tr>
<tr>
<td>eprints.org</td>
<td>University of Southampton</td>
<td>Software to run centralised, discipline-based as well as distributed, institution-based archives of scholarly publications. The software is OAI compliant, i.e. metadata can be harvested from repositories running the software using the OAI metadata harvesting protocol.</td>
</tr>
<tr>
<td>Fedora</td>
<td>Cornell University</td>
<td>An open source digital repository architecture that allows packaging of content and distributed services associated with that content. Fedora supports OAI-PMH requests on content in the repository.</td>
</tr>
<tr>
<td>MARCXML framework</td>
<td>Library of Congress</td>
<td>A suite of tools, stylesheets, guidelines and XML documents to support MARC21 records in the XML environment. Includes Universitytools to support transformation/migration from oai_marc to MARCXML, including an XML schema for MARC21 records.</td>
</tr>
<tr>
<td>my.OAI</td>
<td>my.OAI</td>
<td>Very simple Perl based OAI Harvester</td>
</tr>
<tr>
<td>Tool/Project</td>
<td>Developer/Provider</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>---------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Net::OAI::Harvester</td>
<td>Ed Summers</td>
<td>Net::OAI::Harvester provides an object-oriented client interface to the data found in OAI-PMH repositories (similar to what LWP::UserAgent does for HTTP).</td>
</tr>
<tr>
<td>OAIA</td>
<td>University of Southampton</td>
<td>Based on PERL and MySQL, OAIA is a simple mechanism for providing caching and aggregating of OAI repositories.</td>
</tr>
<tr>
<td>OAI Java Implementation for Linux</td>
<td>University of Illinois, Urbana-Champaign</td>
<td>This is a simple, illustrative implementation of the OAI metadata protocol, using Java. The code is available on Source Forge.</td>
</tr>
<tr>
<td>OAI Implementation for Windows NT/Windows 2000</td>
<td>University of Illinois, Urbana-Champaign</td>
<td>This is a simple, illustrative implementation of the OAI metadata protocol, using Microsoft Windows NT server technologies. The code is available on Source Forge.</td>
</tr>
<tr>
<td>OAIbiblio PHP Data Provider</td>
<td>ibiblio</td>
<td>OAIbiblio is a data provider implementation of the OAI-PMH, version 2.0. Written in a PHP object-oriented manner, this toolkit can be easily customized to communicate with an already existing, multi-table database. Some of the features include: rendered metadata mappings(transformations using separate XSL files, retention of highly expressive storage-level metadata with WDDX, storage-level separate from application &amp; DBMS independence.</td>
</tr>
<tr>
<td>OAICat</td>
<td>OCLC</td>
<td>OAICat is a Java Servlet web application providing an OAI-PMH v2.0 repository framework. The framework can be customized to work with arbitrary data repositories by implementing some Java interfaces. A demonstration implementation is available for download on the OAICat home page.</td>
</tr>
<tr>
<td>OAIHarvester2</td>
<td>OCLC</td>
<td>The OAIHarvester2 Open Source project is a Java application providing a combined OAI-PMH v1.1 and v2.0 harvester framework. A sample harvester application is included, or the included classes can be used within custom Java applications.</td>
</tr>
<tr>
<td>oai-perl library</td>
<td>University of Southampton</td>
<td>A library of PERL language classes that allow the rapid deployment of an OAI compatible interface to an existing web server/database.</td>
</tr>
<tr>
<td>PEAR::OAI</td>
<td>ZZ/OSS Information Networking</td>
<td>A Perl implementation of the OAI-PMH Data Provider which is a PHP class library based on the PEAR classes.</td>
</tr>
<tr>
<td>OAI-PMH Pack</td>
<td>Infrae</td>
<td>Infrae has extended Silva so it allows users to browse and search harvested metadata, further enriching the extensive feature-set of this open source CMS. An organization that uses Silva can thus easily become an OAI-PMH Service Provider. In the process, Infrae also developed a module for accessing OAI-PMH compliant repositories in Python, and developed a sophisticated harvesting and indexing system for using harvested metadata in Zope. These reusable components are designed to be building blocks for other Python or Zope-based applications.</td>
</tr>
<tr>
<td>Tool/Software</td>
<td>Owner</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PEAR::OAI</td>
<td>ZZ/OSS Information</td>
<td>A Perl implementation of the OAI-PMH Data Provider which is a PHP class library based on the PEAR classes.</td>
</tr>
<tr>
<td>Harvesters</td>
<td>Virginia Tech.</td>
<td>Object-oriented harvester class with support for OAI-PMH v1.0, v1.1, and v2.0. Includes sample code to illustrate usage.</td>
</tr>
<tr>
<td>PHP OAI Data Provider</td>
<td>University of</td>
<td>This implementation completely complies to OAI-PMH 2.0, including the support of on-the-fly output compression which may significantly reduce the amount of data being transferred.</td>
</tr>
<tr>
<td>Oldenburg</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rapid Visual OAI Tool</td>
<td>Old Dominion</td>
<td>Rapid Visual OAI Tool (RVOT) can be used to graphically construct a OAI-PMH repository from a collection of files. The records in the original collection can be in any one of the acceptable formats. The formats currently supported are RFC1807, Marc subset &amp; COSATI formats. RVOT helps to define the mapping visually from a native format to oai_dc format, and once this is done the tool can respond to OAI-PMH requests. The tool is self-contained; it comes with a lightweight http server and OAI-PMH request handler and is written in Java. The design of RVOT is such that it can be easily extended to support other metadata formats.</td>
</tr>
<tr>
<td>University</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Static Repository Gateway</td>
<td>LANL</td>
<td>An implementation of a static repository gateway that complies with the specification at <a href="http://www.openarchives.org/OAI/2.0/guidelines-static-repository.htm">http://www.openarchives.org/OAI/2.0/guidelines-static-repository.htm</a></td>
</tr>
<tr>
<td>utf8conditioner</td>
<td>Cornell University</td>
<td>This is a small C program that will either check or ‘fix’ a UTF-8 byte stream. It was designed to be used within an OAI harvester to attempt to remove bad codes from supposedly UTF-8 byte streams so that they can then be parsed using a standard XML parser which would otherwise fail.</td>
</tr>
<tr>
<td>VTOAI OAI-PMH Perl Implementation</td>
<td>Virginia Tech</td>
<td>This toolkit implements the skeleton of the OAI-PMH v2.0 in an object-oriented fashion, thus hiding the details of the protocol from code that is derived from the predefined class.</td>
</tr>
<tr>
<td>XMLFile v2.2</td>
<td>University of Cape Town</td>
<td>This is an OAI-PMH data provider module that operates over a set of XML files that contain the metadata. It is meant to require a minimal of effort while retaining all the flexibility of the OAI protocol.</td>
</tr>
<tr>
<td>ZMARCO</td>
<td>University of Illinois, Urbana-Champaign</td>
<td>ZMARCO is an Open Archive Initiative Protocol for Metadata Harvesting (OAI-PMH) 2.0 compliant data provider. The ‘Z’ in ZMARCO stands for Z39.50; ‘MARC’ stands for MAchine-Readable Cataloging; and the ‘O’ stands for OAI, as in the Open Archives Initiative. ZMARCO allows MARC records which are already available through a Z39.50 server to relatively easily be made available via the OAI-PMH.</td>
</tr>
</tbody>
</table>

4. Summary

This document discusses several tools and software packages that may support interoperability. At least two of them - SimpleServer and ZContent may be used as Z39.50/SRU/SRW gates to DBMS of a particular DAMS. Zebra, Lucene, and Cheshire are information retrieval/indexing systems that use their
own databases and it is not clear how these systems may be used as gates. However, they may serve as backend of THDI Web portal to store indexes for metadata records from various repositories.

Test installations are required to answer questions about connectivity of there tools with particular databases.