EXTENDING OAI-PMH PROTOCOL WITH DYNAMIC SETS DEFINITIONS USING CQL LANGUAGE

Cezary Mazurek
Poznań Supercomputing and Networking Center
Noskowskiego 12/14, 61-704 Poznań, Poland

Marcin Werla
Poznań Supercomputing and Networking Center
Noskowskiego 12/14, 61-704 Poznań, Poland

ABSTRACT

This paper describes the OAI-PMH protocol extension which allows to define dynamic sets of items. Dynamic sets in this context are sets which were not defined in the repository prior to the client request and are defined by the set specification in the request. The main aim of the proposed extension is to decrease the number of records (the amount of data) transferred from the repository to the harvester. Dynamic sets can be harvested with the OAI-PMH protocol just as all typical static OAI sets. Dynamic sets can also be combined with other (dynamic and static) sets as their subsets. This extension provides the OAI-PMH interface clients with the possibility to harvest their custom-defined sets and to search the repository. In the paper we present in details the concept of OAI-PMH dynamic sets and its prototype implementation in the dLibra Digital Library Framework. The implementation of the presented concept is based on coding the dynamic set specifications in OAI-PMH requests with the CQL language. The proposed extension and the idea of dynamic sets are not covered by the OAI-PMH set definition; however, the solution proposed hereby allows repositories extended in such a way to remain OAI-PMH compliant.

KEYWORDS

OAI-PMH, CQL, protocol extensions, interoperability, distributed search, digital library management systems

1. INTRODUCTION

The increasing number of digital repositories (and the increasing amount of digital resources) caused the need for interoperability between those repositories. Unified mechanisms for accessing diverse repositories were necessary to build advanced network services like educational portals, domain search engines, meta-repositories or even grid-based applications (Kosiedowski, 2004). The Santa Fe Convention, which was held in Santa Fe, New Mexico, on October 1999, started a process of creating a new network protocol which is now called The Open Archives Protocol for Metadata Harvesting (OAI-PMH). This protocol was designed as a consistent interface to digital repositories, allowing easy access to the metadata of all stored objects or to a predefined subset. One of the main principles for this protocol design was also its effortless implementation. Therefore it was based on HTML and XML standards on the communication level and the Dublin Core Metadata Element Set was used as the base schema for metadata exchange.

The metadata exchange process based on the OAI-PMH protocol has two main actors: service providers (*harvesters*) and data providers (*repositories*). Service providers use the OAI protocol to access metadata exposed by data providers through the OAI interface. Service providers may obtain the entire metadata of the harvested items (*records*) or only basic information (*headers*). In general the process of metadata retrieval (*harvesting*) consists of subsequent requests issued by a harvester with harvest parameters and responses from a repository containing metadata of harvested items. The OAI protocol has two mechanisms which enable the selective metadata harvesting. The first one is based on the modification date of a repository item (addition, change, deletion). It allows the harvester to harvest only records modified in a specific period of

time and it is a required protocol feature. The second mechanism is optional and uses a criteria of set membership for harvested items. A set is a group of items in a repository and it may have subsets. One item may be a member of any number of sets (including zero).

The current version of the OAI-PMH protocol (ver. 2.0) was published on June 14th, 2002. Since then the protocol has become very popular. Nowadays the official OAI list of registered data providers (http://www.openarchives.org/Register/BrowseSites) consists of over 700 repositories. The University of Illinois OAI-PMH Data Provider Registry (http://gita.grainger.uiuc.edu/registry/) lists more than 1 500 active repositories. And Google lists over 12 000 of such repositories (checked with the following query: *allinurl:?verb=Identify*). There are also multiple network services based on the data harvested with the OAI-PMH protocol (see http://www.openarchives.org/service/listproviders.html for some examples). One of the most popular OAI-based services is OAIster (http://www.oaister.org/), "a union catalogue of digital resources". Currently it has almost 14 million of records harvested from over 905 repositories.

Another well-defined protocol that can be used to access metadata in digital repositories is the SRU protocol (Denenberg, 2007 a). This protocol tries to unify the syntax of search queries used in the Internet and the structure of responses. SRU was designed as a possible replacement for Z39.50 protocol and is based on it in many aspects. SRU has three types of possible operations: "explain", "scan" and "searchRetrieve". The first one can be used to learn about the functionality and the index of a repository. The second one, the "scan", allows to access the index of terms used to describe items in the repository. The "searchRetrieve" is a crucial one. It allows submitting a query in the Contextual Query Language (CQL) (Denenberg, 2007 b) and retrieving the list of items that matched the query.

An OAI-PMH as well as SRU protocol seems to be similar. In general both have the same goal - retrieval of metadata from remote systems. The main difference between the OAI-PMH and the SRU protocols may be described in the following way: "*If a set of data were exposed via SRU as well as OAI-PMH, then SRU would be the tool to use if a person wanted to extract only data crossing predefined sets. OAI-PMH would be more apropos if the person wanted to get everything or predefined subsets of the data.*" (Morgan, 2004).

The concept of dynamic sets specification can be based on features combined from those two protocols. For better understanding of our approach we start with the following scenarios of concurrent usage of those two protocols described in (Sanderson, 2005):

- SRU interfaces to OAI aggregated data - the repository has an SRU interface to the database with metadata harvested via OAI-PMH from multiple OAI repositories.
- OAI interfaces to SRU provided data - the repository has an OAI-PMH interface and OAI requests are transformed on-the-fly into SRU requests and sent to a predefined group of repositories. Data from SRU responses is then merged and transformed to proper OAI response format.
- OAI retrieval of SRU discovered data - the repository has an OAI-PMH interface and an SRU interface. The client submits an SRU request, which results in creation of a set with the SRU request results. The set (optionally coupled with the user identity) is internally passed to the OAI interface of the repository. The user can use the OAI interface to harvest contents of a set previously created via SRU.

The third scenario is interesting when there is a need of dynamic sets harvesting. In this paper we would like to propose an alternative approach – the extension for the OAI-PMH protocol. The main difference is that we eliminate the SRU interface to the repository in the process of data discovery. This extension allows harvesters to define dynamic sets of records and harvest them from OAI repositories. The example use case for such functionality can be a domain portal (a portal dedicated to a specific subject, e.g. quantum physics) based on a content harvested from multiple repositories. The portal continuously harvests a group of repositories, locally processes harvested metadata to find items related to its domain and presents chosen items to its users. Such a portal can harvest only sets related to its domain, but not all repositories may have such sets defined. In this case it must harvest the entire repository and then process all the harvested data. This can be a time-consuming process for both the harvester and harvested repositories. With the extension described in this paper the harvester can limit the number of harvested records by specifying additional criteria that must be matched by those records. Hence a part of the records processing is moved from the client (harvester) to the server (repository) side – such approach should allow to decrease the amount of transferred metadata. Also the implementation of such extension is done on the repository side, so it does not require any modifications in the OAI harvester. That is why it may be widely applied.

The next section describes a mechanism of dynamic collections in the Polish platform of digital libraries. This mechanism is based on the idea which is similar to the proposed extension, and it was the first step of this extension development. The third section contains a detailed description of the extension; the fourth section contains protocol specification conformance issues. Section five contains a description of an experiment conducted to verify the usefulness of the extension. The last section contains a summary and some directions for further works.

2. DYNAMIC COLLECTIONS

One of advanced content services enabled in repositories available in the Polish NREN network PIONIER (Mazurek, 2006 a) is the service of virtual dynamic collections. This type of collection is an additional mechanism for grouping digital objects, apart from static and dynamic collections. Static collections of digital objects are collections statically set up by the repository staff, for example, a collection of articles related to quantum physics. On the other hand, dynamic collections are collections defined by repository users and described as a set of criteria for digital objects. If an object matches such criteria, then it is included in the particular dynamic collection. Such dynamic collections can be seen as preserved user queries and its results. We defined virtual dynamic collections as dynamic collections of objects from multiple repositories (Mazurek, 2006 b). The implementation of virtual dynamic collections in almost twenty repositories available in the PIONIER network is based on the RSS protocol – it is an RSS interface to data harvested with the OAI protocol. The URL for RSS feed contains the dynamic collection criteria and elements of the feed are elements of such collection. The RSS feed itself contains elements matching given criteria that were (according to the OAI harvested data) modified in the last seven days. This approach is intuitive and easy to use for digital library users. Each time they submit a query, in response they receive its results and a link to the RSS feed with a dynamic collection based on that query. Users can subscribe to such a feed with their favorite RSS client and they will be automatically notified about new items that matched the criteria expressed in the query. Unfortunately, this solution cannot be applied in a situation where there is a network service instead of users. It is because the RSS feed:

- does not contain the full metadata of selected items (by default),
- does not contain all items that matched the criteria – only the records from last 7 days,
- does not allow to split the response into multiple "pages", which may be crucial when there are a lot of items that matched the query.

All the above problems may be solved if the OAI-PMH protocol will be used instead of RSS feeds. Such a solution is described in the following chapter.

3. OAI-PMH DYNAMIC SETS - HOW TO...

3.1 ...harvest a dynamic set?

The idea of dynamic sets in the OAI-PMH protocol is an extension of sets defined in the protocol specification (Lagoze, 2002). Sets described in the protocol specification may be created manually by repository maintainers or may be created automatically, for example, from the content type of digital objects (Nelson, 2006). Each set should have at least its specification (i.e. a unique identifier within the repository hierarchy) and a human-readable name. The meaning of set is not imposed by the OAI specification. It is said that a specific group of repositories may agree on a meaning of sets according to their needs. As it was mentioned earlier, each set may have subsets and items in the repository may belong to any number of sets. If an item belongs to a subset, it also belongs to its superset. If an item belongs to a set, the set specification should be included in the OAI header for this item. Empty sets are allowed. A client request for harvesting all items in a specific set may look like this:

```
(1) verb=ListRecords (2) &metadataPrefix=oai_dc (3) &set=SomeSet:SomeSubset
```

It means:

> (1) get the metadata (2) in the Dublin Core schema (3) of all items from *SomeSubset* subset of set *SomeSet*

In response such a request should return an XML list of OAI records from a set specified by *SomeSet:SomeSubset* with their metadata in the Dublin Core schema.

The example mentioned above shows the request for a static set of items. A dynamic set is a set for which a criterion for set membership is defined by the harvester. As the above example shows, the only place where the criteria can be placed without adding additional parameters to the request is the set specification. So the OAI request could look like this:

```
verb=ListRecords&metadataPrefix=oai_dc&set=SomeSet:EncodedCriteria
```

which means that returned items should be from the *SomeSet* and additionally it should match the *EncodedCriteria*. Additionally there should be a special set specification for marking dynamic sets. It is required to properly process user queries in the situation where *SomeSet* accidentally would have a subset with the specification that perfectly matches the *EncodedCriteria*. For example, the real value of *EncodedCriteria* should be *SomeSubset*. In such a case the OAI repository would not be able to determine if:

- the harvester wants to harvest all items from the *SomeSet:SomeSubset* set, or
- the harvester wants to harvest all items from the *SomeSet* set matching the *SomeSubset* criteria.

To avoid such a situation, a special reserved word (e.g. "*criteria*"" in our case) could be used for a dynamic (sub)set specification. In such a case:

- for the query `&set=SomeSet:SomeSubset` - all items from *SomeSet:SomeSubset* will be returned,
- for the query `&set=SomeSet:criteria:SomeSubset` - all items from *SomeSet* matching the criteria *SomeSubset* will be returned,
- for the query `&set=criteria:SomeSet` - all items from the entire repository matching the criteria *SomeSet* will be returned.

3.2. ...encode dynamic set specification?

We would like to propose the CQL language for the criteria encoding. CQL is a query language designed for various information retrieval systems and it is a part of SRU protocol specification. Its syntax is intended to be intuitive and human readable and writable. In general a CQL query consists of a single search clause or multiple clauses and Boolean operators. Each clause may have two forms: single search term (e.g.: "*Albert Einstein*") or a search term with an index and its relation (e.g.: *dc.creator = "Albert Einstein"*). In the second example the phrase *dc.creator* is an index and should be interpreted in the index context. In this example the context is *dc* which means Dublin Core and is one of the predefined contexts. Among other predefined contexts there is also the *marc* context for the MARC 21 standard for bibliographic description. Both of those contexts can be easily connected with the OAI-PMH protocol, because in this protocol there are predefined metadata schemes for both Dublin Core and MARC 21. Additionally both CQL and OAI-PMH allow to define custom metadata schemes, therefore usage of CQL within the OAI-PMH should not restrict the OAI-PMH protocol flexibility. To conform to the restrictions of the OAI set specification, the CQL-based dynamic OAI set specifications should be URL-encoded (e.g.: *dc.creator%3D%22Albert%20Einstein%22*).

3.3. ...notify harvester about dynamic set?

There are three possible approaches to notifying the harvester about repository support for dynamic sets:

- The notification in the response to the *Identify* request - the *Identify* response in the OAI protocol has an optional repeatable element *description*. This element can be used to expose any additional information about the repository. It is often used to show the syntax of identifiers and

the software used as a repository basis. The *description* element for dynamic sets may additionally contain the list of sets supporting dynamic subsets.

- The notification in the response to the *ListSets* request - the *ListSets* response returns all sets available in the given repository. It could additionally return an extra *criteria* subset for each defined set and one *criteria* set at the top repository level. For example:

```
<set>
   <setSpec>criteria</setSpec>
   <setName>(Gateway for dynamic set)</setName>
</set>
<set>
   <setSpec>SomeSet</setSpec>
   <setName>Some set in our repository.</setName>
</set>
<set>
   <setSpec>SomeSet:criteria</setSpec>
<setName>
       Some set in our repository. (Gateway for dynamic set)
</setName>
</set>
<set>
   <setSpec>SomeOtherSet</setSpec>
   <setName>Other set in our repository.</setName>
</set>
```

  The above sequence informs the harvester about the possibility to use dynamic sets on the top level of the repository and as the subsets of *SomeSet*. There is also the *SomeOtherSet* set which does not support dynamic subsets.

- Do not notify – if dynamic sets are not supported, then a try to harvest such sets will fail. The simplest test can be the following: `&set=criteria`. If it returns the "*badArgument*" error, then it means that dynamic sets are not supported by this particular repository. Otherwise it should return *"noRecordsMatch"* error because there was no dynamic set definition given, so no records could be matched to the set definition.

The first and second approach have a very similar functionality, but the implementation of the first approach may require more work on the harvester side – it must be able to parse and understand the information in the additional *description* element. The third approach does not require any additional work besides the dynamic sets implementation, so it is the easiest one from the repository point of view. But it may cause many additional requests from the harvester, especially when the set structure defined in a repository is large. Names of sets in the above example are generated automatically by adding a suffix to the parent set name *(Gateway for dynamic set)*.

4. OAI SPECIFICATION CONFORMANCE

The proposed approach could not be strictly compliant with the current OAI-PMH protocol specification because of the nature of dynamic sets, but we proposed a solution to overcome this problem. There are two compliance problems. The first one is that the repository should list all its sets in the response to the *ListSets* request. This is not possible with dynamic sets because their number is infinite (assuming the infinite length of set/criteria specification). Therefore the only thing that can be done here is to replace all possible dynamic sets with a single *criteria* set which means the possibility to support dynamic sets.

The second problem is that the OAI-PMH specification requires that if a given item belongs to a set, then the set specification should be listed in this item metadata header. With dynamic sets this is questionable. It is not possible to list all dynamic sets to which an item belongs. But again it may be replaced with the listing of

one additional *criteria* subset for each set. Additionally if a harvest is done with a particular dynamic set, then this set can be listed in the items header.

Both problems described above should not cause any problems for a harvester that does not support dynamic sets. Depending on the approach of dynamic sets presentation (see chapter 3.3), such sets may not be visible for this harvester or may look like empty sets. Therefore any OAI-PMH repository extended with the dynamic sets should be still OAI-PMH compatible for all protocol validators.

5. PROTOTYPE IMPLEMENTATION AND TESTS

5.1. Test environment

The main aim of the proposed extension is to decrease the number of records (the amount of data) transferred from the repository to the harvester. This should give a positive effect for both parties – the repository performs a preliminary selection of records based on given criteria and therefore has less data that should be transformed into OAI records and sent. On the other side, the harvester has less data to receive and to process.

To verify if the above aim may be achieved, we set up a test environment based on all Polish OAI-PMH repositories available in the PIONIER network. The OAI-PMH extension described in this paper was implemented as a prototype feature into the dLibra Digital Library Framework (http://dlibra.psnc.pl/). dLibra is a software platform developed by Poznan Supercomputing and Networking Center since 1999. Currently it is used as the software base of almost 20 various digital libraries in Poland. The current list of dLibra installations can be found at http://dlibra.psnc.pl/biblioteki/. Together all dLibra-based digital libraries give free access to over 100 000 of digital objects.

Each dLibra-based digital library has the OAI-PMH interface giving access to the metadata of all digital objects gathered in this library. The dLibra OAI interface supports sets, which are created from collections defined in the digital library. Each collection has one corresponding set in the OAI interface. Collections can have sub collections and those are represented as subsets in the OAI interface. The OAI set specification (set "name") for each collection is required and it is assigned by the digital library administrator in the collection definition process.

As a harvester for the experiment, the "PIONIER Digital Libraries Federation" (PIONIER DLF, http://fbc.pionier.net.pl/) was used. The PIONIER DLF is a new search service based on OAI-PMH harvested data. It allows to search in all Polish OAI-PMH compliant repositories and digital libraries. The PIONIER DLF harvested repositories may be divided into two groups:

- regional digital libraries – digital libraries maintained by a group of institutions, devoted to history and culture of a specific region of Poland,
- institutional repositories – repositories maintained by a single institution, containing digital resources somehow related to it – in most cases resources created in this institution or resources describing the history of the institution.

Such selection of repositories gives a large variety of digital content. This should be a good basis for testing the OAI-PMH protocol extension described in this paper.

5.2. Test procedure and achieved results

During the experiment we made 9 harvests from all repositories registered in the PIONIER DLF. In those harvest we have specified the following dynamic sets (in brackets there are English translations for *original Polish words* used for sets specifications):

- `dc.language` *`eng`* – publications written in English,
- `dc.language` *`ger`* – publications written in German,
- `dc.type` *`podręcznik`* `(handbook)` – publications of type handbook,
- `dc.type` *`rozprawa`* `(thesis)` – publications of type thesis,
- `dc.type` *`czasopismo`* `(magazine)` – publications of type magazine,

- `dc.type` *`gazeta`* `(newspaper)` – publications of type newspaper,
- `dc.subject` *`pedagogika`* `(pedagogy)` – publications about pedagogy,
- `dc.subject` *`chemia`* `(chemistry)` – publications about chemistry.

Those sets were prepared as example dynamic sets which would be useful in a scenario described in section 1 – a some kind of thematic service (portal) utilizing the data harvested from multiple, diverse repositories.

Table 1 below shows the number of repositories and records harvested in the above harvests. The number of harvested repositories is a number of repositories which returned at least one record in the harvest. The first row of the table shows the total number of records harvested when dynamic sets where not used.

Table 1. Number of records and repositories harvested in the dynamic sets experiment.

Query	Harvested number of		Harvested % of overall number of	
	repositories	records	repositories	records
none (all records)	*16*	*93681*	*100,00%*	*100,00%*
dc.language *eng*	13	626	81,25%	0,67%
dc.language *ger*	12	10357	75,00%	11,06%
dc.type *podręcznik* (handbook)	4	104	25,00%	0,11%
dc.type *rozprawa* (thesis)	5	199	31,25%	0,21%
dc.type *czasopismo* (magazine)	16	28163	100,00%	30,06%
dc.type *gazeta* (newspaper)	4	33793	25,00%	36,07%
dc.subject *pedagogika* (pedagogy)	8	130	50,00%	0,14%
dc.subject *chemia* (chemistry)	8	715	50,00%	0,76%
dc.subject *Poznań*	8	2759	50,00%	2,95%

As we can see, the usage of dynamic sets may significantly decrease the amount of data that was harvested and processed by client. In most performed harvests there were repositories that did not have any documents conforming to the given criteria. It is shown in a graphical form in figure 1 below.

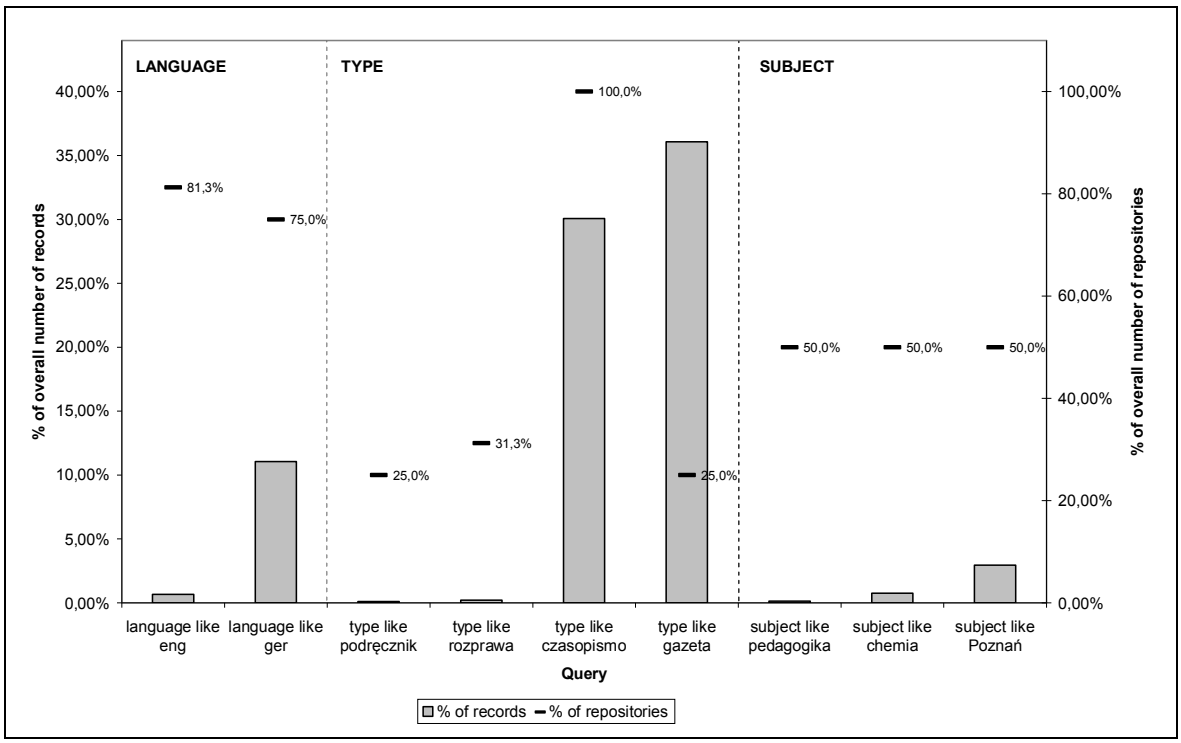

Figure 1. Reduction of the number of harvested records achieved with the proposed OAI-PMH extension.

6. CONCLUSION

In this article we have proposed an extension for the OAI-PMH protocol. This extension adds searching functionality to the OAI protocol. It is based on dynamic sets expressed with CQL. Dynamic sets can be harvested by any OAI-compliant harvester without any modifications. We have also shown that using this extension leads to significant reduction of the amount of harvested items. Apart from improving the performance of OAI harvester, this extension also creates new possibilities, especially for harvesting large and diverse OAI repositories. With this extension OAI harvesters are no longer limited to the predefined sets.

In the near future we will focus on the development of new applications in which new possibilities enabled by the described extension will be used. Those applications (mainly domain portals) will be initially based on Polish digital repositories using dLibra Digital Library Framework and will be deployed in the PIONIER DLF portal.

ACKNOWLEDGEMENT

Work under the research project nr 3 T11C 02330, ``Mechanisms of atomic services for distributed digital libraries'' financed by Polish Ministry of Science and Higher Education.

REFERENCES

R. Denenberg et al, 2007 a. *Contextual Query Language, version 1.2*. http://www.loc.gov/standards/sru/specs/cql.html.

R. Denenberg et al, 2007 b. *SRU (Search/Retrieve via URL), version 1.2*. http://www.loc.gov/standards/sru/.

M. Kosiedowski et al, 2004, Digital library grid scenarios. *In Knowledge-Based Media Analysis for Self-Adaptive and Agile Multi-Media, Proceedings of the European Workshop for the Integration of Knwoledge, Semantics and Digital Media Technology, EWIMT 2004*. London, UK.

C. Lagoze et al, 2002. *The Open Archives Initiative Protocol for Metadata Harvesting*. http://www.openarchives.org/OAI/openarchivesprotocol.html.

R. LeVan, 2006. *SRU and Lucene*. http://staff.oclc.org/~levan/SRU%20and%20Lucene.ppt.

C. Mazurek et al, 2006 a, Distributed digital libraries platform in the PIONIER network. *In Lecture Notes in Computer Science*, Vol. 4172, pp. 488–491.

C. Mazurek et al, 2006 b, Metadata harvesting in regional digital libraries in the PIONIER network. *In Campus-Wide Information Systems*, Vol. 23(4), pp. 241–253.

C. Mazurek et al, 2006 c, Distributed services architecture in dLibra digital library framework. *In Future Digital Library Management Systems (System Architecture & Information Access)*, pp. 26–31.

E. L. Morgan, 2004, An introduction to the Search/Retrieve URL service (SRU). *In Ariadne*, Vol. 40.

M. L. Nelson et al, 2006, Efficient, automatic web resource harvesting. *In Eight ACM International Workshop on Web Information and Data Management (WIDM 2006)*. Arlington, Virginia, USA, pp. 43–50.

R. Sanderson et al, 2005, SRW/U with OAI: Expected and unexpected synergies. *D-Lib Magazine*, Vol. 11.