

ECP-2006-DILI-510003

TELplus

**Guidelines for preparing a Z39.50/SRU target to
enable metadata harvesting**

Deliverable number	<i>D-2.3</i>
Dissemination level	<i>Public</i>
Delivery date	<i>30th of June 2009</i>
Status	<i>V1.0 – Final</i>
Author(s)	<i>Nuno Freire(BNP), Diogo Reis(IST)</i>



eContentplus

This project is funded under the *eContentplus* programme¹,
a multiannual Community programme to make digital content in Europe more accessible, usable and exploitable.

¹ OJ L 79, 24.3.2005, p. 1.

Contents

CONTENTS	3
1 INTRODUCTION	4
1.1 THE CASE FOR HARVESTING METADATA VIA Z39.50/SRU	4
2 INTRODUCTION TO SEARCH & RETRIEVAL PROTOCOLS	6
2.1 Z39.50	6
2.1.1 <i>History</i>	6
2.1.2 <i>Z39.50 Description</i>	6
2.2 SRU/SRW	7
2.2.1 <i>History</i>	7
2.2.2 <i>Description</i>	8
3 RELATED WORK	10
4 METADATA HARVESTING VIA Z39.50/SRU	11
4.1 DEPLOYMENT SCENARIOS	11
4.2 THE CURRENT STATUS OF SRU/Z39.50 USAGE	12
4.3 METADATA HARVESTING EFFICIENCY CONSIDERATIONS	12
5 HARVESTING METHODS	13
5.1 INCREMENTAL HARVESTS	13
5.2 FULL HARVESTS	14
5.2.1 <i>Full harvest by record creation and modification dates</i>	15
5.2.2 <i>Full harvest by identifier export</i>	16
5.2.3 <i>Full harvest by sequential identifier</i>	17
5.2.4 <i>Full harvest by index scan</i>	18
5.3 CHOOSING A HARVESTING METHOD	19
6 CONCLUSION	21
7 REFERENCES	22

1 Introduction

Collections from the national libraries can be made available in The European Library portal via three communication protocols: the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH), Z39.50 or Search/Retrieve via URL (SRU).

The choice of the communication protocol greatly influences the functionalities that the portal can provide to the end-user. Although all three protocols provide a standard for communication between the portal and the libraries' systems, the underlying communication paradigm is substantially different. While OAI-PMH's design allows the portal to harvest all metadata records from the libraries into a central repository, Z39.50 and SRU were designed for remote search and retrieval, therefore metadata records remain only at the data provider.

The choice of the communication paradigm (metadata harvesting or remote search and retrieval) has an enormous impact on the usability of the portal, mainly on search and retrieval, which is an essential functionality of the portal. Having the metadata harvested into a central repository allows the portal to preprocess the metadata in order to provide services for the end-user, and generally improve the user experience. On the other hand, with remote search and retrieval, the functionality that the portal can provide for the metadata collections is limited by the functionality that the underlying communication protocol. This means that the portal can only provide search and retrieval functionality for the collections that are available by SRU or Z39.50. It also has the implication that the portal has to handle each collection independently of the others, thus not allowing the user to see a consolidated view of the results of his query, forcing him to navigate in the results of each collection individually.

The European Library is also preparing to become a library aggregator for Europeana. Europeana was designed from start to follow the metadata harvesting paradigm, with OAI-PMH being the only supported protocol. Only those collections that are harvested into the central metadata repository of the European Library can be made available to Europeana.

The above factors make a strong case for The European Library to pursue the objective of having the totality of the collections from libraries harvested into its central metadata repository.

1.1 The case for harvesting metadata via Z39.50/SRU

This document is the result of a study to evaluate the possibility of performing metadata harvesting on collections that are available via Z39.50 or SRU.

Although many national libraries already have OAI-PMH access implemented, or are working in that direction, some cases exist where an implementation of OAI-PMH would require a major effort, due to several kinds of difficulties in adapting the underlying information systems of the national libraries.

An important factor for OAI-PMH implementation is vendor support. When vendors of library management systems provide an OAI-PMH module for their products, that is usually the choice taken by the libraries. However, even though OAI-PMH was designed to be a simple communication protocol, with a low cost of implementation, many vendors do not provide OAI-PMH module for their products. In these cases, libraries have to look for other solutions, that require more technical knowledge from the library staff.

Libraries can implement OAI-PMH by deploying one of the software solutions made available by The European Library or by using other software available as open source and free of any charges.

Independently of the software in use for the OAI-PMH implementation, an essential component is the one that accesses the metadata records from the catalogue and makes them available to the OAI-PMH software. This typically may take two forms:

- An export tool provided by the library management system to export the metadata records to a file.
- An implemented middleware which accesses the metadata records from the catalogue, typically via a direct connection to the relational database of the library management system.

Some difficulties may arise with any of the above solutions. For example, in some systems the export tools have to be used by a person, therefore the process cannot be automatic. Or connecting directly to the database of the library catalogue may not be at all possible because the database in use is a proprietary solution without an open API.

In these scenarios, a possible solution may be to harvest the metadata records through the Z39.50 server of their catalogue. Unlike OAI-PMH, Z39.50 servers are available for the large majority of library management system vendors, and its usage within libraries is widely spread.

Metadata harvesting done through Z39.50/SRU will certainly present difficulties of its own. Z39.50/SRU were not designed for metadata harvesting, so some functionality required to make the metadata harvesting process efficient and reliable was not included in the protocols design. So, this application of metadata harvesting will likely be a very inefficient process that will place a significant load on the libraries' information systems.

2 Introduction to search & retrieval protocols

This section aims to introduce the search and retrieval protocols.

2.1 Z39.50

The protocol Z39.50¹ is an ANSI/NISO standard for computer-to-computer communication designed to support searching and retrieval of information from remote databases, provided by a server, in a distributed network environment [1]. The architecture of Z39.50 is database-oriented, at a higher level of abstraction than Database Management Systems because the semantics is oriented to classes of databases.

2.1.1 History

The initial efforts in Z39.50 started in the 1970s [2], an experimental protocol as part of the Linked System Project (LSP), with the goal of creating a national bibliographic database in the United States. For that was defined a cross-database searching standard that could perform searches across the major bibliographic databases, which were very homogenous, in organizations such as the Library of Congress, the Online Computer Library Centre (OCLC), and the Research Libraries Information Network.

The following work by the early 1980s was mainly implementation, delaying the creation of a protocol. In 1988, when the standard was established, the result was focused on information retrieval from bibliographic databases and there wasn't a correct implementation.

By the end of the 1980s the community interested in the standard had grown and there were now many libraries with online catalogues, having more classes of databases. NISO released a revised version of Z39.50 in 1992, Version 2. This version had a much greater feedback from people implementing it. Still, a major barrier in its deployment was the definition of presentation layer services.

In 1995 the version 3 of Z39.50 was published. It contained important improvements in performance on fast networks, sorting and access point browsing. It also introduced complex features like extended services and the generalized record syntax.

Z39.50-2003 revises the standard to incorporate clarifications, amendments and corrections and has been endorsed by the Z39.50 Implementers Group (ZIG).

2.1.2 Z39.50 Description

Z39.50 is a connection-oriented protocol that defines interactions between two machines. The current protocol is over TCP/IP. The first version was over X.25, but TCP/IP has a better performance and was used afterwards.

The communication has an initialization phase where client and server negotiate the communication properties (version of the protocol, maximum record size, etc.). The server may require the client to authenticate. Either the server or the client may end the session.

¹ Z39.50 Maintenance Agency Page, <http://www.loc.gov/z3950/agency/>

A server has databases containing records. Each database has a set of access points (indexes) that can be used for searching. This is a much more abstract view of a database than one finds with SQL, for example. One deals only with logical entities based on the kind of information that is stored in the database, not the details of specific database implementations.

One of the basic Z39.50 functions allows the client to transmit a search to the server (a SEARCH request). There are over a hundred search parameters for bibliographic records, in five categories: relation attributes, position attributes, structure attribute, truncation attributes, completeness attributes. The queries have a Boolean structure and may have multiple nesting of search phrases e.g. (phrase1 AND (phrase2 OR phrase3)). Also allowed in the query structure are operands for Restriction and Proximity. The semantics of the SEARCH request in Z39.50 was the basis for SRU/SRW Common Query Language. A search produces a set of records, called a "result set", that are maintained on the server. Records from the result set can be subsequently retrieved by the client using PRESENT requests.

A server can provide progress reports for an active search, or can ask the client for authorization to continue a resource intensive search; a client can abort an active search.

Z39.50 contains facilities for managing and sorting result sets, browsing the values of access points associated with a database, for opening and closing connections, and a general mechanism called "extended services", which is an asynchronous remote procedure call mechanism that the client can use to invoke services on the server, optionally making reference to the contents of a result set as a parameter.

Various groups have developed Z39.50 profiles. Of notice is the Bath Profile¹ that specifies a rigid syntax for bibliographic searches. The maintenance agency has a list of these, but the maintenance and support of the profiles is unclear. This has led to a fragmentation in the implementer community and is a serious problem for the purpose of interoperability.

2.2 SRU/SRW

SRU² is an acronym for Search/Retrieve via URL, a standard search protocol for Internet search queries, utilizing Common Query Language (CQL) – a query syntax for representing queries based on the semantics of Z39.50.

SRW³ (Search Retrieve Web Service) is a variation of SRU, where messages are transmitted with XML over HTTP with SOAP⁴ instead of URL.

SRW and SRU are intended to define a standard form for Internet search queries as well as the structure of the responses. SRW/U was defined by the ZING (Z39.50 International: the Next Generation) Group.

2.2.1 History

With the advent of the Internet, the flaws of Z39.50 started to become apparent. SRW/U was developed with the intention of using: HTTP instead of a telnet connection; a single URL as a request; and an XML response instead of a dialog of commands with some complexity. The

¹ Interoperability Focus: The Bath Profile, <http://www.ukoln.ac.uk/interop-focus/bath/>

² SRU: Search/Retrieve via URL, <http://www.loc.gov/standards/sru/>

³ SRW: Search/Retrieve Web Service, <http://www.loc.gov/standards/sru/srw/>

⁴ SOAP Specifications, <http://www.w3.org/TR/soap/>

main advantage of SRW/U compared to Z39.50 is its simplicity and the use of standards (like HTTP, XML, etc.).

SRU concepts retained from Z39.50¹: Result Sets; Abstract Access points; Abstract Record schemas; Explain; Diagnostics.

The version 1.0 (November 2002) of SRU was an experimental version. The current version, released in February 2004 is 1.1. SRU Version 1.2 is the current version.

2.2.2 Description

Both SRW and SRU have only three operations²:

- explain – requests to learn about the server's database. The server must respond with the location of the database, its content description and protocol features supported by the server.
- scan – list and enumerate the terms in the remote database's index.
- searchRetrieve – the most important operations. It's the mechanism to query the remote database, with the CQL. The queries may be from free text to Boolean operations with nested queries. Some aspects of CQL are optional, but the servers must reply to unsupported requests with diagnostic messages. The result of the query may be returned in several metadata formats (like Dublin Core, RDF, MARCXML, etc.), as specified in the *explain* operation.

CQL is a formal language designed to be human readable, intuitive and with enough potential to express complex queries. It's possible to query using:

1. Boolean logic (ex: book or magazine):
2. numeric comparisons (ex: year > 2006)
3. proximity of words in a document (ex: book prox/distance<=5 magazine)
4. across multiple dimensions (ex: date within "2002 2005")
5. relevance (ex: subject any/relevant "fish frog").

The difference between SRU and SRW is the way messages are encapsulated and transmitted (see Figure 1). SRW is a SOAP-ful Web Service. Messages sent and received are encapsulated with SOAP. Because of this, HTTP is not a necessary transport protocol, although it is generally used for practical reasons. SRW over HTTP requests use the POST method. POST has the advantage of imposing no limitations on the length of arguments. SRU is a REST-ful Web Service, which means operations are encoded as name/value pairs. All operations are transmitted as HTTP GET requests. The results are XML streams, as with SRW, except there is no SOAP envelope. The benefits of SRW over SRU are: better extension support, authentication, and web service compliance (W3C standard).

¹ SRU and Z39.50, <http://www.loc.gov/standards/sru/short-topics.html#z3950>

² SRU: Version 1.2 and Beyond - DLF Spring Forum 2006, <http://www.diglib.org/forums/spring2006/>

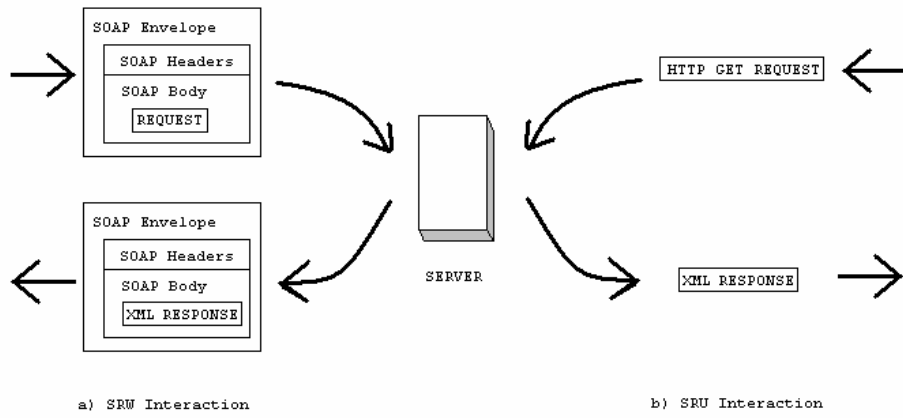


Figure 1 - a) SRW and b) SRU communication with a Server

3 Related work

A similar scenario to that faced by The European Library regarding metadata harvesting by Z39.50, was addressed in the project “Yellow Brick Roads: Building a Digital Shortcut to Statewide Information” in the United States of America, by the Library of the University of Illinois at Urbana-Champaign. This project investigated the feasibility of unified searching across library holdings, digitization projects, and online state government information through use of the OAI-PMH together with the Z39.50 protocol. Z39.50 was not used for search and retrieval, but for metadata harvesting. Their experience is documented in [3].

The project faces a similar scenario that is present nowadays to The European Library, with several libraries that have their catalogues available only by Z39.50. Of particular relevance for The European Library, is their study on the viability of using the Z39.50/OAI Gateway Application Profile in existing Z39.50 servers. The OAI-PMH/Z39.50 Gateway Profile¹ was created for the purpose of creating an appropriate response to OAI-PMH requests layered over the Z39.50 server.

They analyzed if Z39.50 servers deployed in libraries had the necessary features to allow the actual implementation of the OAI-PMH/Z39.50 Gateway Profile. The requirements that this profile demands from the underlying data structures and search mechanisms of the Z39.50 servers were not supported, and the vendors did not have any plans to implement them.

We have no reason to believe that the above conclusion is not applicable to the European libraries. We tested a sample of European Z39.50 servers and they also did not support the full set of required functionalities.

Another work, which is relevant to our work, analyzed the potential usages of OAI-PMH together with SRU [4]. Of particular interest is their analysis of what functionalities a SRU target must provide to allow an OAI-PMH gateway to be built over the SRU interface. It is therefore a similar analysis to that done in the Z39.50/OAI Gateway Profile, but for SRU. It does not present any real-world deployments of the technique, so it does not provide any clues to its applicability in existing SRU servers.

¹ <http://frasier.library.uiuc.edu/research.htm> - During the course of our work, we were not able to access the URL of the site that hosts the application profile.

4 Metadata harvesting via Z39.50/SRU

This section presents the analysis of the available possibilities for performing metadata harvesting via Z39.50/SRU in The European Library.

4.1 Deployment scenarios

In the context of The European Library, the two scenarios have been identified where metadata harvesting via Z39.50/SRU could be used. These two scenarios are aligned with the ongoing work in TELplus for building an OAI-PMH infrastructure for The European Library.

In the first scenario, metadata harvesting via Z39.50/SRU is deployed at the library. That is, the library harvests its own Z39.50 server, and makes the harvested metadata available to The European Library by OAI-PMH. In this scenario, the Z39.50/SRU harvester is a form of middleware between the library management system and the OAI-PMH server of the library, as shown in Figure 2.

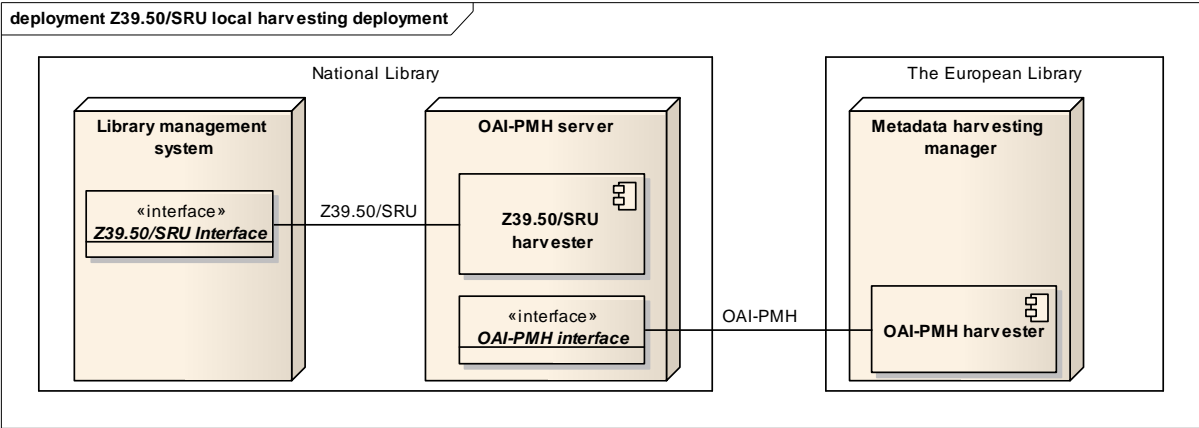


Figure 2 - Deployment of a local Z39.50/SRU harvester

In the second scenario, metadata harvesting via Z39.50/SRU is deployed at The European Library. In this scenario, the Z39.50/SRU harvester will be integrated in the central metadata harvesting infrastructure of The European Library, as shown in Figure 3.

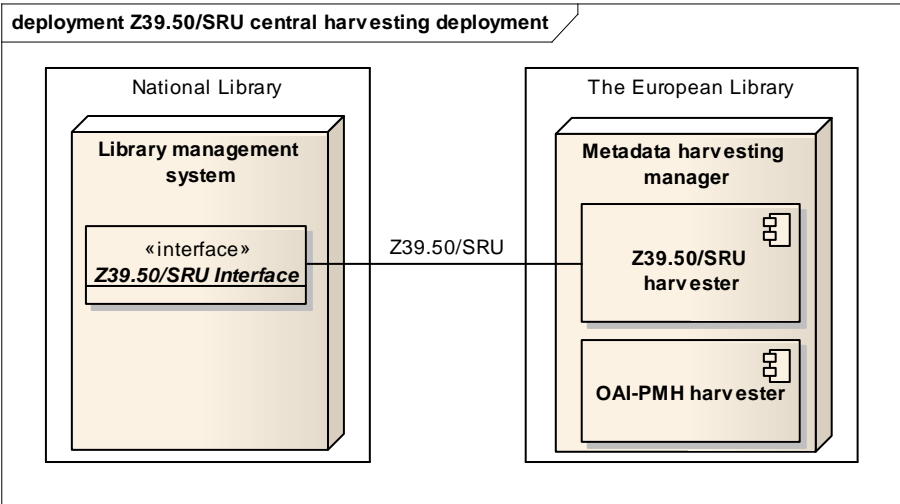


Figure 3 - Deployment of a Z39.50/SRU central harvester

A possible third scenario is to deploy a gateway that translates OAI-PMH requests to Z39.50/SRU requests in real time, as in the Z39.50/OAI Gateway Application Profile. However, as presented in Section 3, it is unlikely that the Z39.50 servers deployed in libraries have the necessary features to allow a successful implementation of a gateway.

4.2 The current status of SRU/Z39.50 usage

Since the start of the TELplus project several libraries of The European Library have moved from Z39.50/SRU to OAI-PMH. Eleven libraries are implementing OAI-PMH in TELplus, and others are implementing it in the context of local projects/activities.

A significant number of libraries still have collections available by Z39.50, representing 27% of the searchable collections of The European Library. In these cases, an implementation of OAI-PMH would require a major effort, due to several kinds of difficulties in adapting the underlying information systems.

However, SRU is currently used only for 3% of the searchable collections of The European Library. This percentage is expected to drop in the short term since these collections originate from four national libraries, and two of them have OAI-PMH implementations in progress. The remaining two libraries have SRU implementations based on Z39.50/SRU gateways, therefore these collections are also available via both SRU and Z39.50.

In this context, supporting metadata harvesting by SRU would provide little added value to The European Library, therefore it should focus mainly on metadata harvesting via Z39.50.

4.3 Metadata harvesting efficiency considerations

Distributed search protocols like Z39.50 and SRU were not designed to fulfill requirements for metadata harvesting. The methods described in Section 5 make it possible to harvest Z39.50/SRU servers, however in some cases the process can be inefficient or computationally expensive.

The harvesting methods can place a heavy load on the Z39.50/SRU targets and/or on the network. Its usage should be tested prior to usage, as it is not possible to predict in advance how the target will perform. The target performance can vary from several factors such as:

- The size of the collection;
- The processing capacity of the hardware;
- The network bandwidth available;
- The network latency;
- The server load from other applications and users of the target;
- The database structure of the target.

The description of the methods in Section 5 will discuss their implications for the overall efficiency of the harvesting process.

The efficiency of the harvesting process can be further affected if performed directly from The European Library to the Z39.50 or SRU targets at the libraries (the scenario 2 of Section 4.1). In this scenario the network latency and bandwidth can make the harvesting process very slow and unreliable. Therefore the application of this scenario should be tested prior to its usage for every target individually.

5 Harvesting methods

Metadata harvesting was not what distributed search protocols were designed for, therefore there is no single standard way on how to perform metadata harvesting. The only work developed to define such standard is the Z39.50/OAI Gateway Application Profile, but its application in real-world scenarios showed that it cannot be applied to the existing Z39.50 servers.

For this reason, this document will describe other techniques that can be used to harvest search targets. These techniques are based on functionalities that are commonly available in Z39.50 servers. They will not provide an ideal harvesting mechanism like OAI-PMH or the Z39.50/OAI Gateway Application Profile. These methods, although allowing metadata harvesting, may in some cases be inefficient or computationally expensive. For this reason we will also discuss their efficiency.

We will start by discussing the requirements for incremental harvest to be possible. Incremental harvests will enable a more efficient way of harvesting, because when synchronization is taking place, only the updates are transferred, as opposed to full harvests that must transfer the complete collection whenever the harvester needs to update its copy of the collection.

When a search target cannot support the requirements for incremental harvest, it will only be possible to use the full harvest techniques.

Regardless of the method used for harvesting, the search target has to provide the metadata records in a format that can be used in The European Library. That is, it must provide full records in MARC format encoded either in ISO 2709¹ or XML (according to the MARCXML² or MarcXchange³ schemas), which can be converted to the TEL Application Profile. Alternatively the target may provide TEL Application Profile records that can be directly used by The European Library.

The description of the methods are focused on Z39.50 and use according terminology, but the same methods can be applied to SRU targets, as both protocols share the same underlying concepts.

5.1 Incremental harvests

Incremental harvests are the preferred method of harvesting, since they only transfer the data that has changed between harvests. The changed records are retrieved by querying the target on the date of creation or modification of the records.

In order to support incremental harvests, the target must meet the following requirements:

- It must support queries by date of creation (Z39.50 Bib-1 use attribute “1011” – “Date/time added”).

¹ http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=41319

² <http://www.loc.gov/standards/marcxml/>

³ <http://www.loc.gov/standards/iso25577/>

- It must support queries by date of last update (Z39.50 Bib-1 use attribute “1012” – “Date/time last modified”).
- It must support relation attributes “less than”, “less than or equal”, “equal”, “greater than” and “greater than or equal” (Z39.50 Bib-1 relation attributes “1”, “2”, “3”, “4” and “5”).
- It must not limit the number of search results.
- The records must contain an identifier, that is unique in the database.
- Deleted records should be retrieved in the search results. This is not mandatory but is recommended.

Figure 4 shows an example harvest, carried out with this method. The harvester starts by sending a search request to the search target, querying for records that were created or modified since the last harvest. It then sends present requests for obtaining the full records, until the whole result set is harvested.

Z39.50 servers that support this method of incremental harvesting, when harvested for the first time, can follow any of the full harvesting methods. The requirement for full harvest by record creation and modification dates are automatically supported by targets that support incremental harvests.

In those cases where the target does not keep track of deleted records, the harvester will not know that the records should be deleted. In order to remove these records from the harvester, it has to perform periodical full harvests in addition to the incremental harvests.

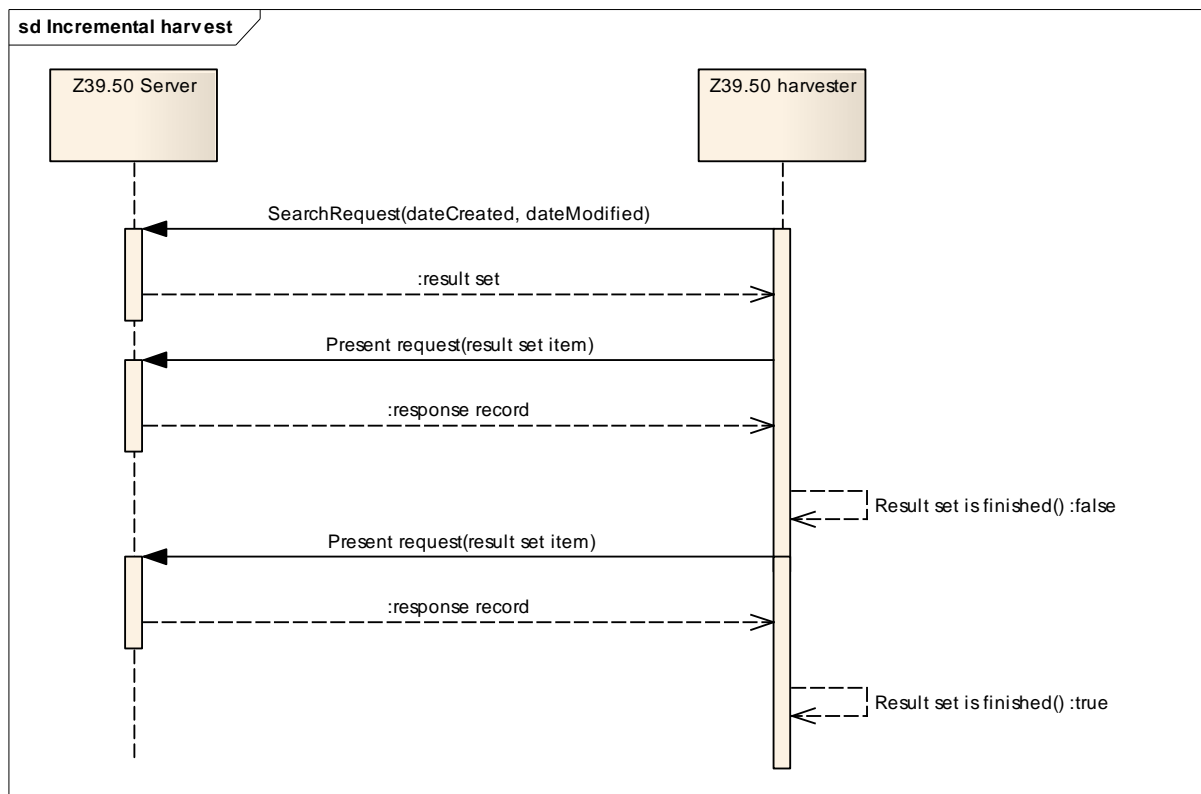


Figure 4 - An incremental harvest by record creation and modification dates

5.2 Full harvests

Full harvest methods have to be used when harvesting a target for the first time or whenever the requirements for incremental harvest cannot be met by the target. These methods can place

a heavy load on the search targets and/or on the network. Its usage should be tested prior to usage, as it is not possible to predict in advance how the target will perform.

This section will describe several methods for conducting full harvests, and will discuss their implications for the efficiency of the harvesting process.

5.2.1 Full harvest by record creation and modification dates

This full harvest method is based on the same functionality of the incremental harvest described in section 5.1, therefore has similar, but lighter, requirements. The records are harvested by querying the target on the date of creation or modification of the records.

The target must support the following requirements:

- It must support queries by date of creation (Z39.50 Bib-1 use attribute “1011” – “Date/time added”).
- It must support queries by date of last update (Z39.50 Bib-1 use attribute “1012” – “Date/time last modified”).
- It must not limit the number of search results.
- The records must contain an identifier, that is unique in the database.

Figure 5 shows an example harvest, carried out with this method. The harvester starts by sending a search request to the search target, querying for records that were created or modified in the earliest date stamp of the database. The earliest date stamp should be configured on the harvester. It then sends present requests for obtaining the full records, until the whole result set of that day is harvested. It then proceeds in the same way for the following days, until the present date is reached, and terminates the harvest.

Z39.50 servers that support the method of incremental harvesting, will also support the requirements for this full harvest method. Therefore, this method should be used when harvesting those targets for the first time, or when full harvests are performed with the purpose of removing deleted records from the harvester.

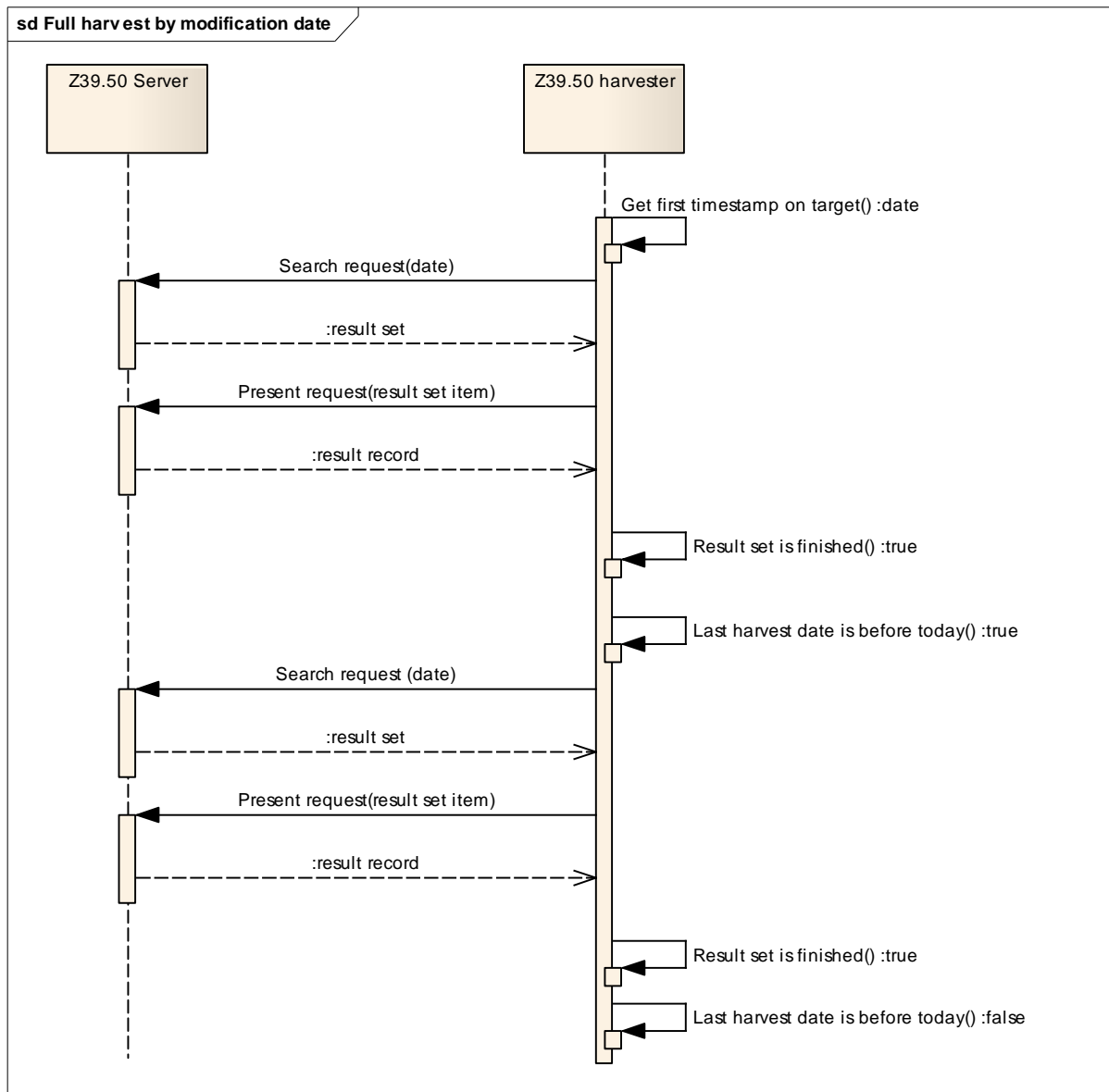


Figure 5 - A full harvest by record creation and modification dates

5.2.2 Full harvest by identifier export

This method requires that the data provider is able to export the complete list of records identifiers of its data set, into a file. That file is then made available to the harvester that will query the target for each of the identifiers individually.

The data provider target must support the following requirement:

- Must be able to export all record identifiers into a file. It has to be accessible by the harvester, either on the file system of the harvester, or by HTTP.

The search target must support the following requirement:

- It must support queries by record identifier (Z39.50 Bib-1 use attribute “12” – “Local number”)

Figure 6 shows an example harvest, carried out with this method. The first step consists in the creation of a file containing the complete list of records identifiers. The file should be a plain text file with one identifier per line.

Libraries can use several methods for exporting the identifiers. The choice will depend on the library management system in use, but typically this can be done by a manual export mechanism or by software program that collects the identifiers via a direct connection to the database of the library management system.

The first step executed by the harvest will consist in accessing the file and reading the identifiers. Access to the file can be provided by the local file system of the harvester or by an URL.

The harvester starts by sending a search request to the search target, with the first identifier, and followed by a present request for the full record. These operations are repeated for all identifiers.

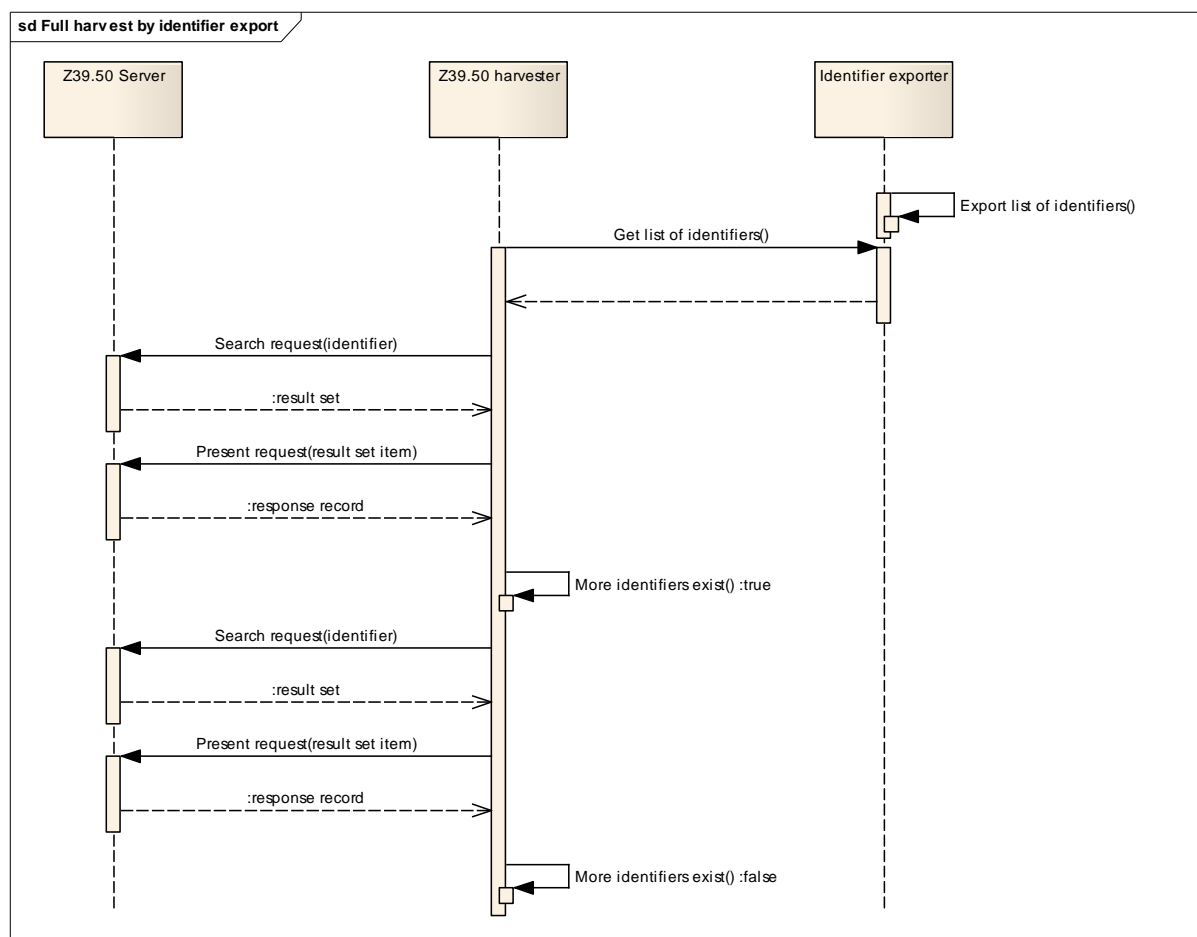


Figure 6 - A full harvest by identifier export

5.2.3 Full harvest by sequential identifier

This method is similar to the previous one, but does not require the initial step of exporting the list of record identifiers. This method is applicable only in systems where records identifiers consist of sequential numbers. In these cases, the harvest will query the target for record identifiers in sequential order until it reaches a maximum. The maximum can be manually configured or be detected by the harvester.

The search target must support the following requirements:

- It must support queries by record identifier (Z39.50 Bib-1 use attribute “12” – “Local number”)

Figure 7 shows an example harvest, carried out with this method. The harvester starts by sending a search request to the search target, with the first identifier. If the result set is not

empty, a present request for the full record is sent and the record is harvested. The harvester then increments the identifier, checks if the maximum identifier was reached, and proceeds by searching for the next identifier or ends the harvest. If the maximum identifier is not known by the harvester, it may stop the harvest when a certain amount of identifiers consecutively failed to be retrieved.

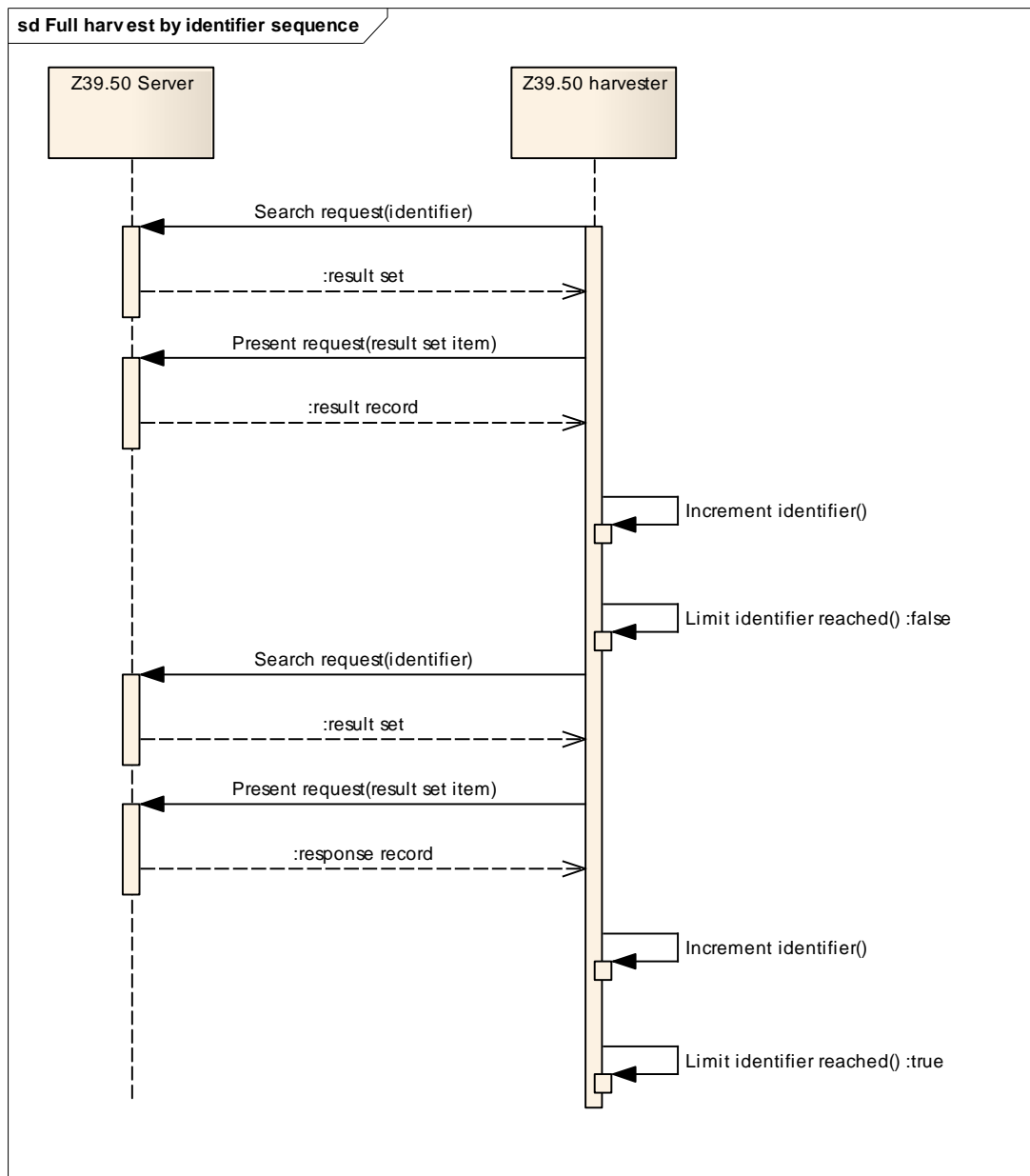


Figure 7 - A full harvest by identifier sequence

5.2.4 Full harvest by index scan

In this method, harvesting is carried out by first executing a harvest of all terms in a predefined index. This initial step is performed by sending successive scan operations to the target, until all terms are obtained. Once all terms are harvested, search requests are sent to the target for each of the terms.

The chosen index should be one where all records of the data set have values on. The harvester will miss any records that don't have any values in the harvested index.

The search target must support the following requirements:

- The target must support scan operations.

- The scan operation must support “step-size” parameter with a value of “0” (zero).
- An index must be available for the scan, and all records must have a value for that index.
- The target must not limit the number of search results.
- The records must contain an identifier, that is unique in the data set, otherwise they may be harvested more than once, or an additional step for detecting and removing duplicates will need to run after harvesting is complete.

Although, theoretically this method allows a target to be harvested, it has several disadvantages:

- It is the less efficient way of harvesting a search target: requires the greatest number of requests being sent to the target; and the same record can be harvested several times.
- It does not guarantee that 100% of the records are harvested: any record that does not contain a value in the index will not be harvested.
- It poses greater requirements to the target, than other methods.
- It is not well supported in Z39.50 implementations: some targets don’t support scan, or only support it with limitations; and it is not always supported by Z39.50 client implementations.

For these reasons the usage of this method is not viable in The European Library. Other methods are more easily supported by search targets. Also, the implementation of this method for The European Library would be more expensive.

5.3 Choosing a harvesting method

When addressing the choice of the harvesting method, a data provider should first try evaluating the possibility of having support in the search target for the requirements of the incremental harvest method. If those cannot be supported a full harvest method should be chosen first on the basis of the requirements that can be fulfilled by the search target. If several methods are possible, then considerations on efficiency should be taken in consideration to make the final decision.

Table 1 provides a quick summary of the methods that can be used when the search target can not fulfill certain requirements.

Regarding the efficiency of the full harvesting methods, the method by record creation/modification date requires more requests to be sent to the target, and therefore will generate a greater load for the target. The methods of harvesting by record identifiers are the most efficient, since all search requests will be lightweight and easily handled by the search target.

	Limits the number of search responses	Does not support search by date stamps	Records do not contain a local identifier	Does not support search by local identifier	Does not support search with relation attributes	A list of identifiers can not be exported
Incremental harvests	✗	✗	✗	✓	✗	✓
Full harvest by record date stamp	✗	✗	✗	✓	✓	✓
Full harvest by identifier export	✓	✓	✓	✗	✓	✗
Full harvest by sequential identifier	✓	✓	✓	✗	✓	✓

Table 1 - Limiting requirements for the harvesting methods

6 Conclusion

This document presented an analysis of metadata harvesting methods, alternative to OAI-PMH, based on the search and retrieval protocols Z39.50 and SRU. Several methods were described and analyzed for their efficiency, but their application in real deployments has to be tested case by case.

Of all the methods described, the combination of incremental harvests, by record creation and modification dates, and full harvests, by one of the record identifier methods, is the most efficient solution if the search target can support the requirements for these methods.

The methods here described will be implemented in the OAI-PMH software under development in TELplus. Therefore these methods will be available for deployment locally at the libraries or centrally in The European Library's metadata harvesting management system.

The implementation will start in July 2009 and will be finished by December 2009. During the development process, the methods will be tested on real Z39.50 targets from European national libraries.

7 References

- [1] W. Moen. The ANSI/NISO Z39.50 Protocol: Information Retrieval in the Information Infrastructure. Available on website: <http://www.cni.org/pub/NISO/docs/Z39.50-brochure/>
- [2] C. Lynch. The Z39.50 Information Retrieval Standard - Part I: A Strategic View of Its Past, Present and Future. Available on website: <http://www.dlib.org/dlib/april97/04lynch.html>
- [3] J. Kaczmarek and C.C. Naun, "A statewide metasearch service using OAI," *Library Hi Tech*, vol. 23, 2005, pp. 576 - 586.
- [4] R. Sanderson , J. Young, and R. LeVan, "SRW/U with OAI: Expected and Unexpected Synergies," *D-Lib Magazine*, vol. 11, Feb. 2005.