METASEARCH XML GATEWAY FOR TOMSK SCIENTIFIC CENTER OF SB RAS

I. Yu. Turchanovsky, O. S. Kolobov, F. E. Tatarsky
Institute of High Current Electronics SB RAS, Tomsk, Russia
e-mail: tur@hcei.tsc.ru, okolobov@hcei.tsc.ru, fet@lib.tpu.ru

Представлен метапоисковый XML шлюз Томского научного центра СО РАН, который применен для доступа к распределенным информационным ресурсам. Описаны детали использования протокола поиска и извлечения информации SRU/SRW. Показана архитектура шлюза.

1. Issue

Metasearch XML Gateway (MXG) [1] is simple search and retrieve information protocol. One of the goals of MXG is to help database providers interoperate with metasearch applications more effectively. The *metasearch* or parallel search, federated search, broadcast search, cross-database search is facility of an application to transmit your search request to several individual search engines, databases and get back results from all search engines queried. Other, ANSI/NISO [2] definition of the metasearch — search and retrieval spanning multiple databases, sources, platforms, protocols, and vendors at one time [3].

This paper describes a *metasearch engine* for Tomsk Scientific Center SB RAS (the center) [13], which is based on NISO-requested SRU/SRW *protocol* [4]. The center has number Z39.50 servers (every Z39.50 database accessible via SRU/SRW protocol [5]), which allow accessing different kinds of information — analytics records, union bibliographic catalog of the center, authority files about subject heading and scientific persons of the center.

The metasearch engine is a calls of service that allow an end user to find content in multiple services with a single search (fig. 1).

2. SRU/SRW protocol

SRU/SRW is XML-based client/server protocol. All messages between client and server are transmitted via HTTP or SOAP mechanism. SRU allows very simple way to send server query request, and SRW released Web Services [12], which supported WSDL description of operations of the protocol.

For detailed description we use a SRU/SRW profile for metasearch engine.

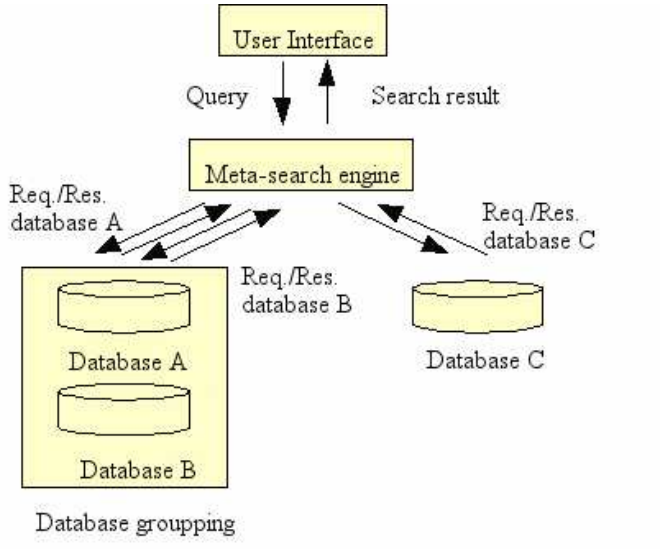

Fig. 1. Metasearch model.

3. SRU/SRW profile for database selecting

Version of the protocol is 1.1. It is the latest official version of the protocol. The subset of supported operations of the protocol are contains explain operation (*explain*) and search and retrieve operation (*searchRetrieve*). The rest operations of the protocol are not part of our profile.

Client and server support query language, *Common Query Language* (CQL), which allows complex search ability for human and automatic systems. There are possibilities of applying a number of attributes for search query — use attribute, relations attribute and others.

There is a problem in using SRU/SRW within the metasearch engine. This protocol does not define a metasearch facility. The metasearch facility can be defined within the profile plus the extension of the protocol. We define the extension for database selecting. The extension of the protocol for *explain* and *searchRetrieve* operation consists of extra element, which is applied to define subset of databases for invoke search query. In other words, we are using an extension of the protocol via *extraRequestData* parameter for each request. The extension of the protocol is used of the XML namespace, `info://srw/extension/634/broadcastSearch-1.0/`. For SRU used an additional HTTP GET parameter *x-info-bs-url*, and for SRW used XML-fragment within *extraRequestData* element of *searchRetriveRequest* (fig. 2).

```
SRU parameter:

x-info-634-bs-url={encoded URL of database}

SRW XML-fragment:

<broadcastSearch xmlns="http://www.arbicon.ru/NS/broadcastSearch/">
<url>{URL of database}</url> </broadcastSearch>
```

Fig. 2. Extension for SRU and SRW cases.

The similar mechanism allows the client to selecting a subset of databases for search request. The main aspect for the metasearch engine is control of resultset. The metasearch engine to creates persistent resultset if there is parameter within SRU/SRW request — *resultSetTTL* (time to live for resultset). It means that client is able to request any records after resultset has been created. In other words, client can use the resultset created like part of the query. This ability of metasearch engine allows to releasing complex query for simple clients.

The metasearch engine is not sensitive from record schemas, because works like a proxy agent between a client and a group of database engines. There is a predefined set of record schemas (Dublin Core [6], MODS [8], MARCXML [7], etc), and each database is able to support one or more record schemas. But not each client of SRU/SRW is able to support full set of record schemas. In cases when client retrieves record from resultset in unsupported record schema, the client will get diagnostic message 66 (*Unknown schema for retrieval*). In these cases the metasearch engine is able to handle request to database engines without sending the request to one.

The metasearch engine generates standard diagnostic messages (used of the namespace — `info://srw/diagnostic/1`), which are divided info several categories: general, CQL, result-set, records, explain, stylesheet, scan.

Also for metasearch engine defined additional diagnostic messages, which used the namespace, `info://srw/634/diagnostic/1`.

4. Architecture of metasearch system

The architecture of metasearch system is extension of *Z39.50 based distributed information search and retrieval system* [9] for SRU/SRW protocol.

There are a few basic components, which defined an architecture of the metasearch system. First component is *service-provider*, which provides search and retrieve information service for client applications. Second component is *content-provider*, which allows access to database

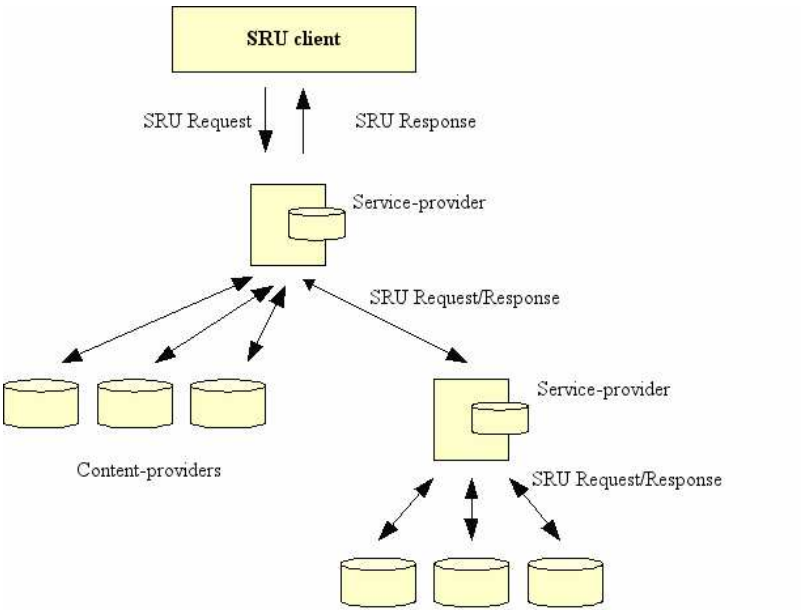

Fig. 3. Architecture of metasearch system.

systems. These architecture define a place for the metasearch engine, service-provider and define *backend databases* like content-providers (fig. 3).

5. Details about the metasearch engine

The metasearch engine is server application written in C++, which is based on YAZ [10], YAZ++ [11] libraries. The server is cross-platform application for Unix/Linux and MS Windows. The metasearch engine support SRU protocol over HTTP, and also SRW over SOAP for backend databases. Maximum number of backend databases is 500. Default record schema is Dublin Core.

6. Conclusion

The metasearch engine is allowed like a general access point for search and retrieve information from heterogenous database systems of the center and it union of bibliographic databases of 5 research libraries of the center. The metasearch engine presents in Internet `http://library.tsc.ru:9001`, which is opened for SRU/SRW client applications.

There is a plan to integrate the metasearch engine within digital repositories of full text documents. This allows to union bibliographics information and metadata of full text documents for search and retrieve operation.

References

[1] NISO Metasearch Initiative — Metasearch XML Gateway Implementation Guide version 0.3. `http://www.niso.org/standards/resources/MI-MXG_v0_3.pdf`

[2] ANSI/NISO. `http://www.niso.org`

[3] Denenberg R. Metasearch and SRU: MXG, the Metasearch XML Gateway. `http://www.loc.gov/standards/sru/march06-meeting/mxg.ppt`

[4] SRU. `http://www.loc.gov/standards/sru`

[5] Kolobov O. Applying SRW/U Protocol to Standard Problems. `http://gpntb.ru/win/inter-events/crimea2005/disk/227.pdf`

[6] Dublin Core. `http://www.dublincore.org/documents/dces/`

[7] MARCXML. `http://www.loc.gov/standards/marcxml/`

[8] MODS. `http://www.loc.gov/standards/mods/`

[9] Жижимов О.Л., Мазов Н.А. Принципы построения распределенных информационных систем на основе протокола Z39.50. Новосибирск: Изд-во ИВТ СО РАН, 2004. 361 с.

[10] YAZ. `http://www.indexdata.dk/yaz/`

[11] YAZ++. `http://www.indexdata.dk/yazpp/`

[12] Web Services. `http://w3.org/ws/`

[13] TSC SB RAS. `http://www.tsc.ru`

Received for publication 19 October 2006