Tomasz Wolniewicz,
Implementing KaRo: The Distributed Catalog of Polish Libraries

from / *aus*:

Union Catalogs at the Crossroad

Edited by

Andrew Lass and Richard E. Quandt

pp. / *S*. 281-294

Erstellt am 31. März 2005

Impressum

Bibliographic information published by Die Deutsche Bibliothek

Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available on the Internet at http://dnb.ddb.de.

Bibliografische Information Der Deutschen Bibliothek

Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.ddb.de abrufbar.

This publication is also openly accessible at the publisher's website. Die Deutsche Bibliothek has archived the electronic publication, which is now permanently available on the archive server of Die Deutsche Bibliothek.

Diese Publikation ist auf der Verlagswebsite ebenfalls open access *verfügbar. Die Deutsche Bibliothek hat die Netzpublikation archiviert. Diese ist dauerhaft auf dem Archivserver Der Deutschen Bibliothek verfügbar.*

Available open access / *open acess* verfügbar:

Hamburg University Press / Hamburg University Press
http://hup.rrz.uni-hamburg.de
Die Deutsche Bibliothek archive server / Archivserver Der Deutschen Bibliothek
http://deposit.ddb.de/

ISBN 3-937816-08-9 (print)

Table of Contents

Part 6 South African Union Catalogs

Chapter 14
Implementing KaRo: The Distributed Catalog of Polish Libraries

Tomasz Wolniewicz

1 Introduction

This chapter was written almost exactly one year after the official launch of the Polish distributed library catalog KaRo. We discuss the functions, limitations and successes of this service, as well as problems and lessons learned for the future and some general observations that can be applied to similar distributed services. The system is under constant development, and the most important features of the new version are described at the end of this chapter.

2 Background

Ever since library catalogs became accessible via the Internet, the need for a coordinated access system to bibliographic data has become apparent. In Poland, the demand for such a system arose from two main directions:

- Reference services, which help users to locate information, often leading to inter-library loans;
- Cataloging, where access to bibliographic data prepared by other libraries drastically reduces cataloging time.

Since the number of library automation systems is rather limited, libraries naturally create groups that use the same software. In Poland, in each of these groups, libraries established their own ways of cooperation for

transferring each others' records. However, things were much more complicated for libraries from different software groups, and even libraries in a single group did not have systems of distributed information service (even if it was technically possible to install such a service).

It should be noted that the views of the present author may be influenced by the fact that he works in a particular library. Nicholas Copernicus University uses the Horizon system, as do some 50 other Polish libraries. These libraries form a very differentiated group, ranging form relatively small to quite large, and from narrowly focused to completely general. This is quite different from the Polish VTLS group, which consists of large academic libraries and is traditionally a leader in standardizing library automation in Poland.

The growing pressure towards a unified service resulted in a successful grant application to The Andrew W. Mellon Foundation for the creation of the Polish Union Catalog (NUKat). The process of defining the role of NUKat turned out to be much more complicated than initially expected. The general assumption that the catalog would contain bibliographical data together with pointers to libraries was never disputed, but there were different approaches to how the data should be entered, which libraries could be represented, etc. One approach was to load the catalog quickly with data from very many libraries, in order to have a wide information service (with a lot of record duplication). A second approach, which was ultimately adopted, was to take every possible precaution against poor quality and duplicated data. The decision to take the second option meant that the widely understood informational role of the catalog will not be realized very soon, which left room for an alternative (distributed) system. Such a project, named KaRo, was launched on July 20, 2001, and is now officially seen as a complementary service to NUKat.

3 KaRo in Practice

The system is available on the Internet at the address http://karo.umk.pl,and provides access to 60 Polish library catalogs (including NUKat) and (after selecting the 'World' option) to nearly 20 additional foreign libraries. The language of KaRo can be switched to Polish or English (although help

screens for the English version were not yet complete at the time of this writing). The user can enter up to three search terms, and select libraries either individually or by predefined groups (university, technical, genera, etc.) or by simply using the 'select all' option. The user also controls the maximum length of time in which the search must be completed, the number of brief results shown on the screen and the type of display of distributed search results.

By limiting the location to one Polish city, the user can turn KaRo into a search service that can specify which library in a given city to go to.

Distributed search results are shown as a list with the number of hits in every selected catalog. In the standard view, this list is sorted into groups in which the search resulted in success, in which nothing was found and some errors appeared and in which a timeout occurred. Each entry in the list leads the user to an individual library where he or she gets access to various details. The first screen for a single library presents results in brief with several records on one screen. By selecting a record, the user is taken to the full view, which shows all relevant fields in the bibliographic record and, if the library provides this information, also holdings details. In the case of journals, the holdings are shown in two levels of detail and can be displayed in ascending or descending order. If the 856 MARC field is filled, the user can get direct access to the electronic source described in this field. In the case of journals, this is usually the link to the electronic version. In the case of the Polish ALEPH libraries, the link leads to the record in the original library OPAC, where some additional information can be found. From the full view, the user can switch to a tabular MARC view. From both full and MARC views the user can save the binary MARC record as a file. The popularity of each view is shown in Table 1

Instead of the standard list of distributed search results, the user can choose to receive the results 'as they come.' In this mode it is not necessary for the entire search to be completed, since first results are available almost immediately, and if they are sufficient, the user can move on much more quickly. The disadvantage of this approach is that the formating is poorer and no sorting into categories is possible. This display format can be also enhanced to provide the function of sending some initial records from each library. This puts a heavier load on the individual library system, and the formating of the result is currently rather unpolished.

Table 1. Percentage Preferences for Views

View type	
Brief view with several (default = 5) short records on one screen	55.6%
Full view in a 'user-friendly format' showing most important fields	35.6%
Full view in MARC format (all fields)	6.7%
Downloading a binary MARC record	2.1%

The initial screen of KaRo also serves as a link to libraries' home pages and to the KaRo single-library mode, in which the user uses KaRo simply as an interface to one library. This has the advantage of providing a well-known tool, rather then having to get accustomed to a new interface for each library system. Unfortunately, it turns out that only 5% of all operations are performed in this mode.

The Users

During one year of service, KaRo has answered over 960,000 queries, by which we mean all accesses to the system that required sending bibliographic data (including a switch of format from standard to MARC). The monthly maximum equaled 124,784 queries in June 2002, with a daily maximum of 7,029 and hourly maximum of 1,349. About 20% of all queries are distributed searches, and the rest correspond to accesses to information delivered by a distributed search. This ratio seems quite stable both in short and long-term observations.

In spite of the very heavy usage, the user base of KaRo is not very large. Over 9,000 different Internet addresses have been seen, but only half of them used the system more then 10 times. The exact distribution of clients is shown in Table 2.

Among the biggest clients, three belong to one public library and in total have used KaRo nearly 90,000 times. There were 514 regular clients who used KaRo more then 50 times and were seen in 5 different months. On a typical day, between 150 and 200 different Internet addresses are observed and over 5,000 queries are answered.

Table 2. Number of Visits and Clients

Number of visits	Number of clients
1 – 9	4,644
10 – 49	3,033
50 – 99	633
100 – 999	728
1,000 – 9,999	146
10,000 – 49,999	11
Total clients	**9195**

Most accesses come from higher educational institutions, public libraries and research institutes, but there is also a significant client base on leased lines supplied by various Internet providers, many of which can be home connections. Some accesses from outside Poland are also seen, but not very often.

KaRo is quite popular among Polish librarians as a cataloging aid. Therefore it may seem surprising that the MARC view is much less popular then the 'user friendly' format view. One of the possible explanations is that in the current version, the user is forced to go through the standard 'full' view, in order to get to the MARC view, and if a new search is performed, the results will always be displayed in the brief view (even if there is only one hit). These are obvious limitations that have already been corrected in the next release under preparation. The reason for the relatively low interest in downloading binary MARC records is probably the difficulty of loading such a record into the local database, especially as the record is saved exactly as it was stored in the supplier database, possibly in a coding format different from that of the local database. Adding a planned translation service to KaRo should help to deal with this problem.

4 Implementation

The idea of using Z39.50 as a basis for a distributed search engine is not new. There a many examples of such systems, of which the Canadian vCuc[1] is probably closest to KaRo. Initiatives like Bath Profile[2] have been established mainly to facilitate a distributed use of Z39.50 by making individual libraries adhere to a common set of standards. There are several features making the KaRo project different from many other Z39.50 based distributed search systems:

- It is a one-man project;
- It is based entirely on free software;
- It requires only minimal cooperation from participating libraries, as all configuration differences are handled inside KaRo; and
- It keeps virtual sessions open indefinitely.

Access to library catalogs is performed via Z39.50 protocol; hence, only libraries providing Z39.50 servers can cooperate with KaRo. Unfortunately, this currently excludes several important libraries.

The core of the system is written in Perl and relies heavily on several publicly available software packages. The main Z39.50 functionality is provided by specialized packages (ZetaPerl in the current version and yaz[3] in new versions), which have been slightly modified. MARC record handling is done by the MARC Perl module, Unicode transliterations are done by the Unicode module, ISBN is handled by the ISBN module, and the Web interface is written with the help of the CGI module. The main user interface is written in PHP and JavaScript.

---

[1] vCuc—Canadian virtual catalog run by the Canadian National Library: http://www.nlc-bnc.ca/8/6/index-e.html.

[2] The Bath Profile: An International Z39.50 Specification for Library Applications and Resource Discovery. http://www.ukoln.ac.uk/interop-focus/bath/1.1/intro.html.

[3] Home page of Indexdata providing the free yaz toolkit: http://www.indexdata.dk.

KaRo is installed on a dedicated two-processor PC running under the Linux/RedHat system. The main program runs continuously, so that only very limited code needs to be started for every connection. This solution complicates the design, but dramatically increases performance and lowers memory consumption. From current observations, it is quite obvious that this system will easily handle a tenfold increase in connections.

Even though Z39.50 is an international standard, individual vendor implementations vary in many small, but important details. In addition, installations in libraries also vary, for instance in the handling of extended characters, the meaning of certain local MARC fields, etc. For these reasons KaRo has quite extensive configuration possibilities, where all these small details are handled. The configuration is much more extensive than in a typical Z39.50 client. Anomalies that need to be taken care of are, for instance, different character encodings in a single record, where the bibliographic part may be encoded differently from the holdings part. Configuration also controls the load, which will be described later.

There are several commercial products available that, at least in theory, can perform the functions of KaRo. Many such systems are in operation throughout the world. Still, there are some good reasons why such a system should be written from scratch:

1. There is enough free software for the realization of various parts of such a system to ensure that the programming task, while non-trivial, is not overwhelming;
2. Writing a system and using software available in source helps to solve some of the problems of closed products. There were cases in which some Z39.50 server implementations were faulty, which led to strange behavior by client software. These problems were overcome by modifying the Z39.50 tools used inside KaRo,
3. With full control over the software, new features can be added easily, but with commercial software, one is limited by the system configuration. In the earlier KaRo versions, one of the software libraries used internally by the system was distributed freely, but in a precompiled form with no access to the source. This created a problem that could not be overcome, which led to the decision to write a dedicated Z39.50 Perl module based on the yaz

software library. Commercial products are expensive and often have license limitations, while Polish libraries have very limited budgets.

Every Web database interface has to implement the notion of a user session. This is particularly important with Z39.50 systems, as a typical access consists of two steps, search and presentation, where presentation of records is based on information provided by the search operation. The Z39.50 server has to keep information obtained from the search operation for future presentation operations. It is obvious that neither the distributed catalog system nor the library Z39.50 server can keep a session indefinitely. It is therefore quite typical for such systems to time out and tell the user to start the session from scratch. Such behavior can be quite irritating, and KaRo produces its Web output in such a way that it can regenerate all information from the output page even if a session has been closed.

5 Load Control

A distributed search system can place a significant load on the resources it uses. At current usage, up to 1,300 queries per hour are serviced. Individual library receives up to 400 queries, but typically not more then 200. Even though that does not seem to be very many, some limits may have already been reached. Here are two main reasons:

- Library consortia typically use a single machine to service many databases. If this happens, the 200 per hour may grow to 2,000 or more, and what is worse, distributed searches hit the server at the same moment with queries to several databases;
- A Z39.50 session normally lasts through the whole of a user's interaction with the database. If the library has a license limit on such connections, there may be a problem both for connections from KaRo and for connections from the local system.

To make the situation less drastic, KaRo can be configured so that within one distributed search it will not send too many queries to a single machine. This lowers the load on the servers, but produces timeouts if the timeout limit is set too low. If KaRo calculates that due to timeout limits and load limits, some libraries will not be reached in time, it immediately sends the

'timed out' report and does not even contact the library database. Since after a distributed search the user will choose a single library from the whole list, it makes no sense to keep all connections hanging; therefore after the distributed search all connections are closed. When a user chooses an individual library, the search is run again (using the KaRo session regeneration mechanism), and this new single session is then held throughout the user interaction. This solution pays some performance penalty, but the overall performance gain and lower load on individual servers make this approach optimal. If a library has very limited license resources, the session timeout may be shortened. This will lower the performance and introduce more operations, but may be a better choice than allowing unused sessions to hang and use valuable licenses.

Currently, there is no overall load control for multiple sessions.

6 Lessons Learned

Running the system for one year, studying statistics and talking to many users has provided some interesting information on user behavior and preferences. We describe some below.

Navigation

KaRo tries to help users by remembering their settings and eliminating unnecessary Z39.50 session initializations. In order to take advantage of this, users must navigate by clicking on the 'new search' link, visible on every page. Unfortunately, this style of navigation is used rather infrequently; it seems that users prefer to navigate using their Web interface 'back' button. A better way of handling users' individual settings should be put in place.

Multiple term searching

Karo allows up to three search terms connected with the logical 'AND.' We have decided not to allow the logical 'OR' operation, since using several terms is mostly done to reduce the number of hits and not to widen the search. The only possible case for the 'OR' would be with subject searches,

where the user is not quite sure of the exact subject classification. Since there were no requests for this functionality and the user interface would have to be a little more complicated, it seemed that it was better to keep only the logical 'AND.'

About 80% of all distributed searches use a single term. 55% of these are title searches, 28% author, 11% ISBN and only 2% subject. Two-term searches account for 17.5%; 95% of these are a combination of author and title, 2% of publisher and title. Only 1.5% of all searches use three terms, half of them a combination of author, title and publisher.

Taking into account the fact that KaRo is quite heavily used for cataloging, it is rather surprising that the ISBN search is quite low. Perhaps librarians search for a similar record and then make modifications.

The very low number of subject searches may be due to the fact that the results obtained will not be meaningful without some form of consolidation of results. Consolidation would require downloading of results from all libraries, and especially in the case of subject searches it could be quite a large task. Another problem is that a unified system of subject cataloging is not yet fully implemented in Poland. An interesting example of subject searches in medical libraries is described later.

Search target selection

Analysis of how users act shows that about half of all distributed searches are performed by selecting all libraries on the list. On the one hand, library directors are very much in favour of KaRo and support it in every possible way; on the other hand, they are concerned with the load it may generate on their systems. One of their requests is to eliminate the possibility of selecting all libraries with one mouse click. When a user profile that permits selections to be remembered is put in place, this automatic selection will be eliminated.

User understanding of the interface

Even though much care was taken to make the interface as self-explanatory as possible and help pages are available for every user screen, there are signals that users have problems understanding that timeouts may be due to

too low timeout limits that they can control. Similarly, the low popularity of using KaRo in the single-library mode may come from the fact that only a few users have read the documentation or have experimented with clicking the link representing an individual library.

KaRo as a back-end system

An interesting use of KaRo was made by Piotr Krzyżaniak, who has set up a WWW interface to Medical Subject Headings (MeSH®). In this system, when users locate a subject heading they are interested in, they can start a distributed search of medical libraries (made by a behind-the-scenes call to KaRo).

7 KaRo and NUKat

As we have explained before, KaRo complements the Polish Union Catalog NUKat. It is expected that the catalogs of the main Polish libraries will be loaded one after another into NUKat with duplicate elimination. At this stage, it would make no sense to search these libraries in distributed fashion, when much better quality results can be obtained by searching NUKat. A more difficult situation will arise if only part of the catalog is loaded. Then, in order to get full results, the local library catalog will have to be searched as well, and we will get duplicate hits for those items that are loaded into NUKat. Distributed searches should still be used when the scope is limited to a certain city or library type. Currently, with a standard distributed search, a user is presented with results in a form of a list of libraries along with a number of hits for each of them. When NUKat is also searched, it appears as another library, but the meaning of results is different, as the exact location of the book can only be known after reading the NUKat record. Unfortunately, duplicated information may be received this way, since some books reported in the NUKat search may also appear in results obtained from a direct library search. The only way to eliminate this possible duplication would be to collect all records found in NUKat and analyze them. This would put additional burdens on both KaRo and NUKat servers and would probably be impractical.

It is quite possible that when the NUKat database becomes quite large, most users will access it directly, and the interest in KaRo will disappear. Such a situation will certainly arise with searches made for cataloging purposes, which is quite natural, since the main goal of NUKat is to speed up cataloging and improve its quality. Searching for rare books will probably be quite useful for much longer. At this moment there seem to be numerous reasons to keep KaRo running and develop it further.

8 Future Work

New features already implemented

We have already mentioned some obvious problems with the current implementation, and indicated that some have been already fixed in the new version.

From the KaRo home page, one can access the experimental version currently under development. There are some major differences between the current stable version and the one under development. The most important is the change of the underlying Z39.50 tools, from ZetaPerl to an in-house module based on yaz.[4] Yaz is under constant development, which guarantees that new features can be added to KaRo in the future In addition, yaz is distributed in the source format and can be modified, for instance to handle servers that do not adhere to the Z39.50 standard in every detail. This change from ZetaPerl to yaz is, fortunately, quite transparent to the user.

One other 'invisible' change is the ability to search and present data in a single network operation, which improves performance. There are also three visible changes:

1. It is possible to save any selection of libraries, so that when one accesses the system again, the selection checkboxes next to the libraries are automatically checked. The option to select all libraries with one click has been switched off, as requested by system librarians. These two

[4] Website of Indexdata providing the free yaz toolkit: http://www.indexdata.dk.

changes together should significantly lower a number of unnecessarily wide distributed searches.

2. It is possible to select the preferred view type to be either standard bibliographic, bibliographic with holdings or MARC. In addition, whenever a search returns only one hit, the preferred full view is used instead of the brief one. This should be a big help for librarians searching with an ISBN number for a single record. In such a case, setting the MARC view as the preferred one will allow the librarian to get this MARC display directly after clicking on one of the libraries visible on the distributed search result list. Of course, one can always redisplay a record in another view.
3. In the list of distributed search results, a small icon next to the record displays the individual result in another window. This allows the user to keep the list of results in one window and easily change libraries to view various possibilities. This feature is in a very experimental phase and currently may introduce some disruption to the system.

Plans for KaRo Version 2

The most important new feature of KaRo V. 2 will be the introduction of individual user profiles. Within a profile, a user will be able to save

1. libraries to be visible on the KaRo list
2. libraries to be initially selected
3. preferred settings of timeouts, number of records per page, bibliographic view
4. preferred search fields.

There will be an option for copying profiles to help establish a common core of profiles for all users of some group. There is no plan to store user identities (names) in the profile. Anyone will be able to create an individual entry. The use of a password will be necessary only when changing the profile.

KaRo V. 2 will have a translator of binary MARC to local character encoding. The setting will also be a part of the user profile.

One important internal change will be added: support for storing the Z39.50 configuration in an LDAP directory. The new configuration will have more load limiting parameters, and KaRo will control the overall load on local systems by counting the number of open sessions, searches in progress, etc. For each library or multiple library server, it will be possible to set load limits which free local systems from unwanted searches. This solution may, in some cases, impair KaRo performance, but this is certainly a better option than forcing a library to withdraw from KaRo altogether.