Proceedings of the International Conference on Distributed Library Information Systems

Editors
Zora Konjović
Dušan Surla

Publisher
Group for Information Technologies
Novi Sad, Bul. Kralja Petra I 13
gint@eunet.yu

Cover
Mirjana Isakov

Printing
ABM Ekonomik, Novi Sad

Circulation
400

Printing costs for this publication were covered by the European Commission through the TEMPUS project JEP 16114.

TEMPUS JEP 16114

International Conference on Distributed Library Information Systems

Ohrid, June 1-6, 2004

Conference Proceedings

Editors
Zora Konjović
Dušan Surla

Novi Sad 2004.

CIP – Cataloguing in Publication
Matica Srpska Library, Novi Sad

021.63:004.4(082)

INTERNATIONAL Conference on Distributed Library Information Systems (2004 ; Ohrid)
Conference Proceedings / International Conference on Distributed Library Information Systems, Ohrid, June 1-6, 2004 ; editors Zora Konjović, Dušan Surla. – Novi Sad : Group for Information Technologies, 2004 (Novi Sad : ABM Ekonomik). – X, 283 str. : ilustr. ; 24 cm

Tiraž 400. – List of Participants: str. 281-283.

ISBN 86-7444-009-6

a) Bibliotečko informacioni sistemi – Zbornici

Foreword

This publication is the result achieved through a part of the activities carried out within the frame of the TEMPUS Project JEP 16114-2001: *Web Based Inter-University Library Network*. The main goal of the Project itself was to establish a library network based on the software system BISIS version 3, consisting initially of the consortium members:

1. Fachhochschule Merseburg, University of Applied Sciences, Merseburg, Germany, Project contractor;
2. University of Novi Sad, Novi Sad, Serbia and Montenegro, Project coordinator;
3. Aristotle University of Thessaloniki, Thessaloniki, Greece, Project partner;
4. Institute of Informatics, Faculty of Natural Sciences and Mathematics, University "Ss. Cyril and Methodius", Skopje, FYR Macedonia, Project partner;
5. Faculty of Technical Sciences, University of Novi Sad, Novi Sad, Serbia and Montenegro, Project partner;
6. Faculty of Sciences and Mathematics, University of Novi Sad, Novi Sad, Serbia and Montenegro, Project partner;
7. Group for Information Technologies, Novi Sad, Serbia and Montenegro, Project partner.

One of the important activities planned within the frame of the Project was the *International Conference on Distributed Library Information Systems*. The Conference was held in Ohrid, FYR Macedonia from June 1st till June 6th , 2004. Representatives of all Consortium members, library and IT experts, attended the Conference. Along with the participants from Consortium members' institutions, several distinguished external experts in library management and library related IT from United Kingdom, Slovenia, FYR Macedonia, and Serbia attended the Conference. Contracting and coordinating institutions, with approval of all consortium members and EU Commission, invited the participants of the Tempus project *Building Cooperative Academic Library Network in Serbia* who also attended the Conference. The total number of participants was 40.

The scope of the presentations given at the Conference targets the problems and solutions for distributed library information systems. The total number of the presented papers, whose authors are librarians, experts in library management and IT experts, is 21. All papers presented at the Conference are published in these Proceedings.

The papers are grouped in three groups. The first group comprises invited papers. These papers deal with general problems related to library management automation, planning and organization of the library services based on the contemporary information technologies, and applications of the modern information technologies to quality control of the records in library databases. The second group of the papers presents the software solutions applied to web based inter-university library network. The detailed presentations of the software system BISIS version 3, as well as the solution to the networked digital library of theses and dissertations of the University of Novi Sad are given. The third group comprises the papers presenting achievements related to applications of automation and information technologies to several libraries of the consortium members. The following libraries were presented:

1. Library of the Fachhochschule Merseburg, Merseburg, Germany
2. National and University Library "St. Clement Ohridski", Skopje, FYR Macedonia
3. Library of Faculty of Humanities and Social Sciences, University of Novi Sad, Novi Sad, Serbia and Montenegro

In the end, the publication contains the list of participants at the Conference.

As the members of the Project team and editors of these Proceedings, we would like to express our gratitude to all participants in the Project, authors of the presented papers and all those attending the Conference. We express the gratitude to Prof. Dr Dušan Petrovački, Secretary of the Provincial Secretariat for Science and Technological Development of Autonomous Province of Vojvodina, for the financial support given by the Secretariat to the development of the Networked Digital Library of Theses and Dissertations. We also would like to thank to Prof. Dr Charalampos Fragkakis, representative of Aristotle University of Thessaloniki for his contribution to the realization of the Project. Dr Joachim Reus, Dr Bernhart Gottschalk and Mrs. Renate Kawohl-Gottschalk did an excellent work in administrative management of the Project and we are grateful for their efforts.

On behalf of all Project participants we would like to express special gratitude to our contractor Fachhochschule Merseburg. Fachhochschule Merseburg provided an unselfish help and support to the Project realization. Therefore, we would like to express our particular thanks to Prof. Dr. rer. nat. habil. Heinz W. Zwanziger, Rector of the Fachhochschule and Prof. Dr Georg Hillrichs, Dean of the Department of Informatics and Applied Natural Sciences.

Very special gratitude we owe to Prof. Dr Ivan Seder, contractor representative and Prof. Dr Margita Kon-Popovska, representative of the University "Ss. Cyril and Methodius", Skopje for the large and valuable contributions to the realization of the Project, efforts made in organizing the Conference and an unselfish and friendly support.

All expenditures related to the preparation of the Conference, travel and stay costs for the participants and the expenditures for printing and publishing these Proceedings are covered by the budget of the Tempus Project JEP 16114-2001: *Web Based Inter-University Library Network* by the European Commission. On behalf of all Project participants and all Conference participants we would like to express the gratitude for this support.

Novi Sad, 2004.

Editors

Table of Contents

Invited Papers

National and International Standards for Library Automation

Alan Hopkinson
Head of Library Systems
Middlesex University, London, UK
A.Hopkinson@mdx.ac.uk

1. Introduction

Librarians like standards compared with most other professions. For example, you will find few librarians who do not agree with the desirability of one type of standard used very much in libraries, classification schemes. You may find librarians who like to develop their own schemes though these are fewer than used to be the case. You will certainly find librarians in charge of classifying who develop their own amendments to existing schemes, but basically librarians like to pay lip-service to standards.

Some authorities suggest that standards stifle innovation [1]. But without a platform supported by a core of standards, many information-based activities would not even begin. The question in every profession is to what level to take standards. The library and information profession is predominantly public sector based and is not as competitive as even the organizations in which the libraries are based. For example, in the United Kingdom, small businesses can usually rely on a public library for reference material and most public libraries will give such information over the phone. There is little room for rivalry between different local authorities since they cover different areas. Universities are more likely in Britain at any rate to be rival institutions and university libraries do not want to support and indeed are not funded to support other university's students; nevertheless there are many agreements whereby students and staff can visit each other's institutions to read material that is not readily available elsewhere. So this spirit of mutual cooperation has had an effect on universities.

2. Standard Building Blocks

The standard building blocks which libraries use have been developed outside the profession. Many standards are not really standards but conventions such as the way words are spelt. Many have developed over centuries. When computers were introduced there was a need for standards to enable computers to communicate with each other and these were developed in the computing industry. Some were developed by the leaders in the field such as IBM. Others were developed by national standards bodies, such as ANSI, the American National Standards Institute or by the International Organization for Standardization in Geneva to which national standards bodies belong. The most famous standard for communication between computers is ASCII, now an international standard [2]. This was derived from the coding used in telexes. Punctuation, letters of the alphabet, numbers etc are represented by combinations of binary characters which make up a range from 0 to 255 (originally this was only to 127).

This is a very interesting standard with which to begin for many reasons particularly in this part of the world. The forerunners of the standards first included only upper case and not lower case letters. You will see that early library catalogue printouts have no lower case letters, they are all in capitals. The codes were very much American if not English language biased so it was not until the codes were extended from 128 to 256 combinations that it became possible to include diacritics as accented characters are popularly called. Initially there was a character set for western European. Each set was called a code page. As time has gone on other code pages have been set up for other languages. Finally this has led to UNICODE [3] which consists of a mechanism of setting up 'large' code pages which include many languages.

3. Record Structures

Given the basic building blocks, we can begin to build computer systems which will serve the functions that libraries need. Library automation began in universities in the early 1960s and by the end of the decade the US Library of Congress and the British National Bibliography had started the MARC programme [4]. These were the days of tape as far as computers were concerned. Computer storage, the equivalent of floppy disks or hard disks in computers today was only under development, and most data was stored outside the computer on magnetic tapes. So the MARC standards were made compatible with the way data were stored on magnetic tapes and given a record structure akin to that of magnetic tape. The data elements found on a catalogue card were allocated three-digit identifiers known as tags. The Library of Congress in conjunction with the American Standards bodies attempted to make this into an American standard but it was not accepted as the scientific information industry wanted to develop similar standards for its abstracting and information services which would have been subtly different. Instead only the record structure was confirmed as a standard [5] and was later adopted as an international standard ISO 2709 [6]. In the first edition of the

American standard the MARC tags were listed for illustrative purposes and were not regarded as part of the standard.

MARC which is not officially a standard illustrates the different players in the standardization activity. What do we mean by official standards? The standards adopted by ISO and its member bodies in different countries. Of course there are proprietary standards which is what we call standards which arise from their adoption by a large and influential manufacturer such as IBM or the now defunct DEC (Digital Equipment Company). In particular fields such as ours there are other players. MARC was developed by national libraries which are the equivalent of IBM in terms of influence in our field. The Library of Congress began the development of MARC in 1966 as a means of circulating catalogue records around libraries in the United States. The British National Bibliography, later absorbed into the British Library joined in 1967 and made some improvements to MARC which the US never accepted. So henceforward there were two formats, US MARC and UK MARC, until 4 June 2004 when the British Library will cease to use UK MARC as it completes the implementation of its new system [7]. Since other libraries have not changed they will provide UK MARC records for up to three years, converting them from MARC21 as US MARC is called today. Other countries adopted their own national formats based usually on the US or UK practice. In the late 1970s it was felt in the international library community that there should be an international format into which every national library should convert its records so under the auspices of IFLA UNIMARC was developed, a manifestation of this format is now COBISS MARC (COMARC). We referred to the different practices between libraries and scientific information providers. This came to a head in the holding in 1978 of the International Seminar on Bibliographic Exchange Formats entitled 'Towards a Common Bibliographic Exchange Format?' [8]. This conference resulted in a new format known as the Common Communication Format (CCF) [9] which was developed under the auspices of UNESCO. Immediately after the meeting mentioned above, a meeting of an ISO working group took place to begin work on a data element directory and this has been the catalyst for other standards [10].

4. Data Element Definitions

In those days library standards were needed most in the catalogue record. Even holdings were not governed by standards as this was not an area where libraries were exchanging data. In any case, to achieve standard records which could be exchanged, MARC alone was not sufficient. A number of other standards had been adopted as international standards. There is the set of standard numbers, ISBN [11], ISSN [12] etc which are international standards. Standard abbreviations have been developed as standards [13]. Additionally to support a standard catalogue record across systems, the series of standards called ISBDs was established based on ISBD(G) [14], also by IFLA, and these were incorporated into cataloguing codes such as AACR [15]. Countries of Eastern Europe wanted to adopt them but IFLA had no authority as a standards making body so they tried to establish it as an ISO standard. This led to problems because ISO standards have to be vetted by committees and these were made of different people with different interests from those who had been instrumental I developing the ISBDs so since IFLA would countenance no changes to ISBD, the idea of it being an ISO standards had to be dropped. So library standards remained supported by unofficial standards and the world stabilised until the development of microcomputers.

5. Microcomputers

The introduction of microcomputers made great changes to the world of library automation for two reasons. Large databases could be stored on a desktop computer so institutions much smaller than the national libraries and other large libraries using mainframe computers began to import data into their system. Organisations in developing countries could set up catalogues on PCs using freely available software such as UNESCO's CDS/ISIS software. Secondly microcomputers began to be used as clients to mainframe systems, allowing processing to be done locally.

Standards were developed for applications such as inter-library loan to enable ILL messages to be sent between different systems. Recently this standard has been feeding into a revision of the data element directory mentioned above.

The large number of library catalogues available for searching spurred on the development of a new standard to enable cross catalogue searching. This was developed by NISO the US standards body dealing with libraries known as Z39.50 [16]. It was later adopted as ISO 23950. It prescribes identifiers for indexes, rules for searching via these indexes and methods of returning the results of these searches. The implementation of the standard by software used to search library catalogues enables a user to search across many systems with different architectures. However, there are no strict definitions of how the indexes are to be generated. Because most systems are based on MARC and ISBD-influenced cataloguing codes there is a certain amount of consistency in the data being searched. But a personal author index for example could be created in many different ways and any searches mad across catalogues will usually not be as effective as if individual searches were made across each.

If you search across catalogues you will find material available only in one library or in libraries other than one's own. If that is the case you may wish to obtain the materials. Some librarians are worried that this ability to search many catalogues with relative ease will increase inter-library loan to an unaffordable level. How can costs of inter-library loans be reduced? One way would be by allowing borrowers to visit other libraries to borrow material they could not find at their home library. A standard has been developed by NISO for this a purpose, known as NCIP, NISO Circulation Protocol [17]: this may be adopted as an ISO in the near future. This enables borrower details to be read from other systems and loans permitted to them in consortia arrangements. It is a parallel system to Z39.50 but for circulation systems to interact with each other.

5. Book Trade Standards

We have mentioned briefly two standards which are used by the book trade, the ISBN and ISSN. The ISBN is currently 10 digits but will be brought in line with the European Article Number and become 13 digits in 2007. This is to increase 'the range of numbers and to bring it in line with other commercial systems.

For ordering, the book trade uses a standard called EDIFACT [18] used by other sectors of commerce for ordering their commodities. Library systems are now implementing orders and invoices via EDIFACT. This is looked after in the UK by BIC, Book Industry Communication, which is supported by a consortium of the Publishers Association, the British Library, CILIP (formerly the Library Association) and the Booksellers Association.

Because of the prevalence of the XML standard mark-up language, the book trade have sought a standard for their interchange of data using XML and have developed ONIX [19]. This is a standard format that publishers can use to distribute electronic information about their books to wholesale, electronic and retail booksellers, other publishers, and anyone else involved in the sale of books.

6. Future Standards: RFID

We have looked at current standards. To give you an idea of how standards bodies work, it is interesting to look at what is happening with RFID. This means Radio Frequency Identifier and it is the system that will replace bar codes in the future. It consists of a chip or tag attached inside a book and which has encoded in it the bar code. In some systems it also has encoded a byte which indicates whether it is issued to go out of the library or not. The data are read by a radio receiver.

This is a developing technology and it has much potential. Standards have been developed outside the library world where warehousing is already using the technology, to specify the frequency of the transmission. Other standards indicate the position of the codes on the chip and the codification to be used. What is needed now is a specification of the kinds of data that should go on a tag used for library circulation control and the format of the data. In 2001 at an ISO meeting it was suggested by the Danish Standards body that there should be a working party to consider these issues. In the event there were not enough countries willing to contribute to this exercise. At the time this might have been controversial since the book trade were very interested in each book holding a chip containing a standard number, the ISBN, followed by a consecutive number which would have made a unique number which libraries as well as booksellers could have used. In the event this was not realised since the cost of tags about 50 euros is too expensive for booksellers to fit in each book when they would have little use for it. Libraries however use the tags repeatedly since books are loaned repeatedly. In May 2003 it was decided to keep an eye on the situation in the event that the work might need to be resumed. In 2002 there was one library in the UK using this technology. In 2003, three new libraries came on board. In 2004 at least three libraries have decided to use RFID. If there is no standard then any library wishing to switch from one system supplier to another might discover that this could not be done without rewriting the data to the tags for all the books in the library, hence the need for a standard.

7. Recap of Standards Bodies

The review of these standards has indicated who is involved in the development of library standards. ISO is the main approval body for standards and after this come the national standard bodies. ISO is a large organization and is divided into technical committees. Technical Committee 46 is the one that deals with libraries and it is called Information and Documentation. This committee has been active for over 50 years [20]. The most active national standards body in this field is NISO in the US. They are interested in NCIP but not in RFID in libraries so NCIP has gone ahead as an American standard. Open URL is another standard to enable the persistence of URLs which NISO has supported [21]. NISO works very closely with the Library of Congress and other sectors of the US Library Community. In the UK, BSI works in this area but generally does not propose standards but has to approve every ISO standard for UK use and can vote against standard proposed by other country standards bodies.

Other bodies maintain library standards. MARC is a standard and the Library of Congress maintains the format with the help of various committees which include representatives of the British Library. The Library of Congress also maintain language codes and look after the maintenance of Z39.50 on behalf of NISO.

Classification schemes also have to be maintained as new concepts develop as well as different ways of viewing certain subjects. Library of Congress Subject Headings are maintained by the Library of Congress [22]. The Dewey Decimal Classification Scheme is maintained by OCLC, the US equivalent of COBISS. UDC is maintained by the UDC Consortium of which BSI is a member, since they publish the English Edition.

IFLA maintains the ISBDs and the UNIMARC standard assisted by the Permanent UNIMARC Committee which consists mostly of representatives of UNIMARC using national libraries or other institutions. IFLA has also over the years contributed very much to research and development in the field. Among the latest is a review of the purpose of bibliographic records which is feeding into the development of the ISBDs and UNIMARC [23].

Book trade standards are maintained in Europe by BIC who act on behalf of the European Editeur Group.

7. Conclusion

This paper has attempted to give an overview of the standards involved in library automation, and the organizations that work on them. Institutions in any country may join IFLA and there is a reasonably active membership in this region. For example, the National Library in Belgrade has recently prepared translations of ISBDs and UNIMARC into Serbian. It is hoped to have a representative from COBISS on the Permanent UNIMARC Committee.

It is much harder to influence the activities of ISO since it depends on the national member body being active and this costs money. Even in the UK we had to fight hard to participate in the ISO TC 46 group that deals with the standards mentioned in this paper. However, NISO do have international experts on their committees and since they develop many of the standards that ultimately become international standards this is one way in which other specialists can become involved.

References

[1] Espina, M.I.; Markman, G.D. "A conceptual assessment of tradeoffs between technological innovation and technological standards" *in: Proceedings of the 2000 IEEE Engineering Management Society. EMS – 2000*. Piscataway, NJ: IEEE, 2000 p.581-5

[2] Information technology – ISO 7-bit coded character set for information interchange. Geneva, ISO, 1991 (ISO/IEC 636: 1991)

[3] The Unicode Consortium's The Unicode Standard, Version 4.0.1 is defined by: *The Unicode Standard, Version 4.0* Reading, MA, Addison-Wesley, 2003, as amended by the website *Unicode 4.0.1* http://www.unicode.org/versions/Unicode4.0.1/

[4] Gredley, Ellen and Hopkinson, Alan. *Exchanging bibliographic data: MARC and other international formats*. Washington: A.L.A., 1990

[5] ANSI/NISO. *Information Interchange Format*. Washington: NISO, 2001 (Z39.2 - 1994 [R2001])

[6] *Format for bibliographic information interchange on magnetic tape.* Geneva: ISO, 1981 (ISO 2709-1981) now called *Format for information interchange*. ISO: Geneva, 1996

[7] The British Library, MARC 21 and the Integrated Library System. [Press release] London: British Library, 2004 http://www.bl.uk

[8] Towards a common bibliographic exchange format? : International Symposium on Bibliographic Exchange Formats : proceedings. Budapest, OMKDK, 1978

[9] *CCF: The Common Communication Format*. Paris: UNESCO, 1984. This has since been published as two volumes: *CCF/B : the Common Communication Format for Bibliographic Information*. Paris: UNESCO, 1992; *CCF/F : the Common Communication Format for Factual Information.* Paris: UNESCO, 1992

[10] *Bibliographic data element directory*. Geneva: ISO, 1988-2000. *Part 1: Interloan applications* (ISO 8459-1:1988); *Bibliographic data element directory Part 2: Acquisitions applications* (ISO 8459-2:1992); *Part 3: Information retrieval applications* (ISO 8459-3:1994); *Part 4: Circulation applications* (ISO 8459-4:1998); *Part 5: Data elements for the exchange of cataloguing and metadata* (ISO CD 8459-5)

[11] *International Standard Book Numbering (ISBN).* 3rd ed. Geneva: ISO, 1992 (ISO 2108-1992)

[12] *International Standard Serial Number (ISSN).* 3rd ed. Geneva: ISO, 1998 (ISO 3297-1998)

[13] *List of Serial Title Word Abbreviations.* Paris: ISSN Centre, 1999

[14] *ISBD(G): General International Standard Bibliographic Description: Annotated Text.* Rev. ed. München : K.G. Saur, 1992 (*UBCIM Publications. New Series*, 6)

[15] *Anglo-American cataloguing rules*; prepared by the American Library Association, the Library of Congress, the Library Association, and the Canadian Library Association. Chicago: American Library Association, 1967. Revisions have taken place at frequent intervals, the latest in 1998 with minor amendments in 2002

[16] *Information Retrieval (Z39.50): Application Service Definition and Protocol Specification* Bethesda, MD: NISO, 1995. (ANSI/NISO Z39.50-1995). Also available as an international standard: *Information retrieval (Z39.50) Application service definition and protocol specification*. Geneva: ISO, 1998 (ISO 23950)

[17] ANSI/NISO. *Circulation Interchange*. Washington: NISO, 2002 (Z39.83 – 2002)

[18] UN/EDIFACT Working Group. *United Nations Rules for Electronic Data Interchange for Administration, Commerce and Transport.* New York: UN, 1995- Available at: http://www.unece.org/trade/untdid/welcome.htm

[19] *ONIX Product Information Standards* [website]; maintained by EDItEUR jointly with Book Industry Communication... London: BIC, 2001. http://www.editeur.org/onix.html

[20] Richardson, George. *The first 50 years of ISO/TC 46: a short history.* Berlin, ISO/TC 46 Secretariat, 1997

[21] ANSI/NISO. *The Open URL framework for context sensitive services: draft.* Washington: NISO, 2001 (Z39.88 – 200X)

[22] *Library of Congress Subject Headings.* 24th ed. Washington, DC:, Library of Congress, 2001

[23] *Functional requirements for bibliographic records: final report of the IFLA Study Group on the Functional Requirements for Bibliographic Records*. Munich: K. G. Saur, 1998 (UBCIM Publications New Series, 19)

Overview of the Development of the Library Information System BISIS

Dušan Surla[1], Zora Konjović[2]
[1]Faculty of Sciences and Mathematics, University of Novi Sad
[2]Faculty of Technical Sciences, University of Novi Sad
{surla,ftn_zora}@uns.ns.ac.yu

Abstract. The library software system BISIS is being developed at University of Novi Sad since 1993. The first version of the system has been in use in libraries since 1996, while the third version is currently in exploitation. Moreover, new versions of the system are being developed.

1. Introduction

The Ministry of Science and Technology of the Republic of Serbia has financed the project [1] which defines the requirements of the library network (LN) of the System of Scientific and Technological Information of Serbia (SSTIS). The same ministry funded the project *System of Information Forming and Searching* from 1993 to 1996. The project has been realised by joint work of five institutions for science and research.

Research results of this project are given in the monograph [2]. The chapter 6 presents results of the sub-project *Design and Implementation of the Sole System for Documents Indexing and Searching* which was realised in «Mihajlo Pupin» institute in Belgrade. The chapter 8 shows the results of the sub-project *Design and Implementing of Library Subsystem of SSTIS* which was carried out at the Faculty of Natural Sciences and Mathematics in Novi Sad. The research conducted during these two subprojects resulted in the software system for library management, named BISIS ver. 1.0. The system is based on the UNIMARC format [3].

The development of the BISIS system was carried on after 1996 at the University of Novi Sad. The work resulted in the BISIS ver. 2.0 and the BISIS ver. 3.0, and the new versions are currently being made. [4] contains the instruction manual for use of the currently used version 3 and the list of published papers concerning the work on the system development (140 references by 41 authors). The most important research portions are grouped within subject matters of MSc. and Ph.D. [5-15].

2. Library Network Requirements

The goal of the project [1] is establishing the basic elements of the development of the computerised library network regarding hardware and software, as well as standards to be complied with when designing and acquiring equipment, producing bibliographic information, i.e. databases of bibliographical records. This project contains the following definition of the term 'library network'. A library network, being an infrastructure part of the SSTIS, is a system of interconnected library information and documentation institutions with considerable funds of scientific and expert references which, through cooperation and with the help of information services and built communication network, supply users with library information, as well as primary documents, in a direct, fast and easy way. The tasks of LN SSTIS performed locally are also given, together with those being carried out on the information service level.

The tasks of LN SSTIS being performed locally are:

- Local support for shared cataloguing,
- On-line search of local databases,
- OPAC,
- Acquisition,
- Circulation,
- Library statistics,
- Check of serials being complete,
- Local support for interlibrary loans,
- Local issuing secondary publications,
- Local segment of coordinated acquisition.

Tasks of LN SSTIS available on the information service level:

- Shared cataloguing of primary documents,
- Central catalogues maintenance,
- On-line search,
- Interlibrary loan,
- Coordinating acquisition of primary documents,
- Issuing secondary publications,
- Information dissemination,
- Directory service,
- Accessing foreign information sources.

The system BISIS has been developed so that the aforementioned tasks of LN SSTIS be realised.

3. Versions of the Software System BISIS

The following terms have been defined in the chapter 8 of the monograph [2]: *document*, *record* and *bibliographical material*. By a document it is meant a source bibliographical unit (a book, a magazine, etc.). A record represents a data set about a document, while a set of records stands for bibliographical material. The same monograph also defines the following functions of the Library information system:

- Defining standards for processing and searching,
- Creating bibliographical material,
- Library reporting and documenting,
- User search,
- Use of bibliographical material.

Defining standards for processing and searching facilitates description of the structure of individual standards and their representation in the database, thus enabling performing basic BISIS functions. Standards for bibliographic material maintenance and records structuring, librarian working environment and catalogues and reports formats are separately defined. The UNIMARC format is adopted as a basis for data exchange and structuring Database of bibliographical material in the BISIS system. The definition of the standard for bibliographical material maintenance is supposed to facilitate the description of the UNIMARC format structure and its representaion in the database in order to provide a mechanism for performing the remaining functions of the system in compliance with the adopted standard. Defining the librarian working environment should allow listing librarians, description of various document types and their representation in the database. Defining catalogues and reports formats is supposed to enable abstract description of the structure of an arbitrary report and catalogue about documents being formed out of the BISIS database, and representing that structure in the database. Structuring search results, librarian reports and documents catalogues would be performed in this way, as well as results of user queries which are results of database search induced by librarians requests and users queries.

Creating bibliographical material means documents processing and structuring records according to the adopted standard using the parameters of concrete processing based on the models of documents processing defined in advance.

Library reporting and documenting enables forming particular library reports and catalogues about documents. Forming reports and catalogues is performed on the basis of Database of bibliographical material and Database of standards for bibliographical documents processing.

User search provides for user queries processing and query results structuring based on the contents of the Database of bibliographical material. Structuring query results is done on the basis of predefined structures, content and display formats which are given in the Database of standards for bibliographical procesing.

Use of bibliographical material facilitates listing library users, as well as following the use of the library fund, reserving and taking documents out. Therefore it is necessary to provide the database of library users in addition to the database of bibliographical material.

3.1. BISIS version 1.0

The software system BISIS ver 1.0 is implemented in the C programming language, and db_VISTA is used as the database management system. Indexing and searching of bibliographical records is realised using the text server developed at the «Mihajlo Pupin» institute in Belgrade. The UNIMARC format is adopted for processing of bibliographical material. This segment is realised at the Faculty of Natural Sciences and Mathematics in Novi Sad. This version of the application defines the following databases: the Database of UNIMARC format, the Database of the librarian working environment, the Database of bibliographical material and the Database of library material use. These databases have been used also in the following versions of the system with necessary changes being made.

This version of the system supports the following functions: Defining standards for bibliographical material maintenance and records structuring, Defining librarian working environment, Creating bibliographical material, Library reporting and documenting and User search.

The specification of the information requirements of this version has been produced using the methodology described in the chapter 4 of the monograph [2] which is based on the Structural system analysis and Entity relationship diagram. The methodology in question has been applied using the *ARTIST CASE* tool which was developed at the Faculty of Organisational Science in Belgrade.

3.2. BISIS version 2.0 and version 3.0

The software system BISIS ver 2.0 has been implemented in the *Oracle* environment and the *Java* programming language. The *Oracle* products *Oracle Server* ver 8.0.3 or later and the text server *ConText Cartridge* ver 2.3.6 or later, which supports *Unicode*, have been used. The system is provided for work in the Internet, i.e Intranet environment. The user interface is implemented in the form of a *Java* applet. This version supports all the functions supported by the previous version.

The main difference between the 2.0 and 3.0 versions is the text server for indexing and searching bibliographical records. The 3.0 version features its own purposefully developed text server for indexing and searching bibliographical records in the UNIMARC format. Its fundamental characteristics are: being dedicated which results in the better performance, three-layered architecture, using the *Java* platform and independence from the relational database management system used. The *Unicode* standard support has been implemented consistently throughout the whole system BISIS ver. 3.0.

The functions supported by this version are: Defining standards for bibliographical material maintenance and records structuring, Defining librarian working environment, Creating bibliographical material, Library reporting and documenting, User search and Library material use.

System modelling has been done using the *Unified modelling language* (UML). It has been implemented as an Internet application in the *Java* environment.

4. Conclusion

The Library software system BISIS is being developed since 1993 and several versions have been released so far. The first version of the system came into use in libraries in 1996, while the third version of the system is currently being exploited. A part of the BISIS system is being realised within the TEMPUS project named *Web-Based Interuniversity Library Network* (JEP 16114). Further research concerning development of this system is currently underway at the University of Novi Sad. The presently pursued interests are: development of the system for digital libraries; development of an XML-based library system; multimedia bibliographical documents search, application of agent technology to bibliographical records search.

References

[1] Project *Plan of Serbia Library Network* Development, Ministry of Science and Technological Development of the Republic of Serbia, Belgrade 1992 (project manager prof. Dušan Surla, Ph.D.) (in Serbian).

[2] Monograph *Creating and Searching Databases in the System of Scientific and Technological Information of the Republic of Serbia*, editor Branislav Lazarević, Belgrade, 1996 (in Serbian)

[3] *UNIMARC manual: bibliographic format/ International Federation of Library Associations and Institutions*, IFLA Universal Bibliographic Control and International MARC Programme, New Providence, London 1994.

[4] D. Surla, Z. Konjović, et al. *Instruction Manual for Use of Library Software System BISIS ver. 3*, Group for IT Novi Sad 2003. (in Serbian)

[5] Tričković, I., *Use of Petri Networks for Specifying Dynamics of Information Systems*, M.Sc. Thesis, Faculty of Natural Sciences and Mathematics, Department of Mathematics, Novi Sad, 1997. (in Serbian)

[6] Holo, I., *Command Language of Library Information System*, M.Sc. Thesis, Faculty of Natural Sciences and Mathematics, Department of Mathematics, Novi Sad, 1998. (in Serbian)

[7] Milosavljević, B., *Text Server for UNIMARC Records*, M.Sc. Thesis, Faculty of Technical Science, Novi Sad, 1998. (in Serbian) http://diglib.ns.ac.yu/ndltd/docs/set1/ndltd6/textsrv.pdf

[8] Savić, N., *System for Generating Catalogue Cards in Library InformationSystem*, M.Sc. Thesis, Faculty of Organisational Sciences, Belgrade, 1998. (in Serbian)

[9] Todorić, D., *Shared Cataloguing and User Search of Bibliographical Data*, M.Sc. Thesis, Faculty of Natural Sciences and Mathematics, Department of Mathematics, Novi Sad, 1998. (in Serbian).

[10] Vidaković, M., *Object Oriented Specification and Implementation of User Interface for Library System*, M.Sc. Thesis, Faculty of Technical Science, Novi Sad, 1998. (in Serbian)

[11] Vulić, T., *Reporting and Documenting in Library Information System*, M.Sc. Thesis, Faculty of Natural Sciences and Mathematics, Department of Mathematics, Novi Sad, 19987. (in Serbian)

[12] Pupovac, B., *Modelling and Implementation of Library Information System Commands*, M.Sc. Thesis, Faculty of Natural Sciences and Mathematics, Department of Mathematics, Novi Sad, 2000. (in Serbian)

[13] Zeremski, M., *Modelling UNIMARC Format in XML Technology*, M.Sc. Thesis, Faculty of Natural Sciences and Mathematics, Department of Mathematics, Novi Sad, 2002. (in Serbian)

[14] Milosavljević, B., *Extensible Multimedia Information Retrieval System,* Ph.D. Dissertation, Faculty of Technical Sciences, 2003. (in Serbian)

[15] Vidaković, M., *Extensible Agent Environment Based on Java Technology*, Ph.D. Dissertation, Faculty of Technical Sciences, 2003. (in Serbian)

Implementation of Quality Control System of XML Bibliographic Records

Gordana Budimir
Institute of Information Science IZUM, Maribor, Slovenia
gordana_budimir@hotmail.com

Abstract. Bibliographic record control is one of the most important functions of library information systems, primarily of systems that support bibliographic record exchange. This paper describes systematization of verifications of UNIMARC bibliographic records that should be performed in order to determine record quality. In the BISIS library system, the XML format for presentation of UNIMARC bibliographic records has been defined with the XML schema that enables modeling of most of systematized bibliographic record verifications. Modeling of other verifications, which cannot be modeled with XML schema, is performed with XSLT and XPath specifications. So defined, XML schema and XSLT specification present the basis of quality control system of XML bibliographic records. This system is implemented in the Java programming environment, with ready tools for validation, parsing and transformation of XML documents. The implementation is independent from bibliographic format, which enables control of bibliographic records in other bibliographic formats if they have equivalent XML schema and XSLT specification defined.

1. Introduction

Quality control of bibliographic records is one of the most important functions of library information systems, as the quality of bibliographic record processing in different library practice functions (such as receiving, saving, searching, presentation or exchanging of bibliographic data) depends on it. For example, results of bibliographic data retrieval depend on the data included in bibliographic records; it is of importance whether records contain all the data required (title, author, year of publication, publisher, etc.) that can be searched on.

Bibliographic data can be entered into a library system through a user interface, by bibliographic record exchange or by the conversion of bibliographic data from some existent bibliographic database. The quality of received bibliographic records varies; they may contain deviations from the rules of the bibliographic format implemented in the system. Thus, errors occurring in records have to be corrected to prevent saving syntactically and semantically incorrect records to local library systems. Correction of as many errors as only possible contributes to the increase in record quality.

In view of this, it is required to accomplish some verification of records, concerning syntax and semantic rules of a bibliographic format. These verifications should be able to detect errors in cataloguing of library materials and warn cataloguers about the missing data or inadequate contents in some record parts. However, cataloguers will still be responsible for bibliographic record quality, as only experienced cataloguers are able to find all the mistakes in a record. Without their knowledge, errors in a record may be found only if an algorithm for adequate verification is defined.

The first problem that appears here is how to define verifications of bibliographic records that can discover the majority of errors in records and enable the best records control. The second problem is the implementation of some system for bibliographic record control that should capture all of these verifications. In library systems, this problem of bibliographic record control is solved in different ways, depending on the bibliographic format and the system implementation environment. Usually, the implementation of a particular library function includes some programming code for verification of some bibliographic record constraints. This paper shows how this problem of bibliographic record control may be universally solved by the quality control system of XML bibliographic records, implemented in the Java environment, independently from the bibliographic format. This system is based on XML (*eXtensible Markup Language*) [1] specifications: validation XML schema (*XML Schema Language*) [2,3,4], XSLT (*XSL Transformation*) [5] and XPath (*XML Path Language*) [6] specifications for additional verification of XML bibliographic records. These XML specifications are developed for UNIMARC (*Universal Machine Readable Cataloguing*) [7] bibliographic records used within the BISIS system [8,9]. Equivalent XML schema and XSLT specification can be modeled for bibliographic record control according to some other bibliographic formats.

2. Bibliographic Record Control

The structure of a bibliographic record and the contents of bibliographic fields, subfields and indicators are defined by the bibliographic format and cataloguing rules, such as ISBD (*International Standard Bibliographic Description*) [10] or AACR (*Anglo-American Cataloguing Rules*) [11]. Bibliographic record control has to verify the structure and contents of records, concerning the syntax and semantic rules of a particular bibliographic format and cataloguing rules. Thus, verifications that should be accomplished for bibliographic record control may be divided into the group of structure verifications and the group of contents verifications. They depend on the type of library materials (monographs, serials, articles, etc.) as record structure and contents depend on them, too.

Additionally, these verifications may be divided, according to the number of fields, subfields and indicators, into:

- singular verifications that check the structure or contents of one single field, subfield or indicator, independently from the appearance, structure and contents of other fields, subfields and indicators,
- cross verifications that check structures or contents of some semantically connected fields, subfields or indicators.

Singular verifications check the next concepts of bibliographic format definition:

- existence of fields, subfields and indicators in a record,
- obligatoriness of fields in a record and of subfields in a field,
- repeatability of fields in a record and subfields in a field,
- indicators typicality,
- fields secondarity (fields that appear in subfield `$1` of fields `421`, `423` or `469`),
- coding of subfield contents,
- length of subfield contents, and
- additional concepts of some subfield contents, such as verification of date or of ISBN and ISSN numbers.

Cross verifications check the structure and contents of fields, subfields and indicators, concerning the structure or contents of other fields, subfields or indicators, or concerning the library material type. For example, cross verifications may verify if the year of publication in subfield `100$d` is greater or equal to the year in subfield `100$c`. All such constraints are defined by the bibliographic format and certain cataloguing rules. Cross verifications of bibliographic records may be divided into four groups of verifications (Table I).

Errors in records that can be found by means of these verifications may be divided into three levels depending on theirs importance:

- fatal errors, that must be corrected before saving a record,
- warning errors, that may be corrected in some circumstances,
- information errors that not need to be corrected.

Group	*Verification*	*Example*
I	verifications that depend on library material type	Hierarchical level for articles is 2.
II	verifications that depend on appearance order of subfields in a field and of fields in a record	If subfield $c exists in field 210, subfield $a must appears before subfield $c.
III	verifications that depend on the structure or contents of other fields, subfields or indicators in the same record	All fields 327 must have the same values of indicators.
IV	verifications that depend on the structure or contents of other fields, subfields or indicators in upper-level records in the same bibliographic base	Subfield 464$1 appears only as a link to records for monograph publications.

Table 1. Groups of cross verifications

All these singular and cross verifications of bibliographic records, according to certain bibliographic format, may be described in tables as shown in [12] for UNIMARC bibliographic records.

3. Modeling of Bibliographic Record Control

If bibliographic records are represented with XML format, XML schema of this XML format can be used for bibliographic record validation. The characteristics of bibliographic records that may be checked with this schema depend on naming of XML elements and their structure [13]. In the BISIS system, the format of XML bibliographic records is defined in such a way that the root element is `record`, elements corresponding to fields are `fxxx`, where `xxx` is field identifier, elements corresponding to subfields are `sfx`, where `x` is subfield identifier, and elements corresponding to first and second indicators are `ind1` and `ind2`. Example 1 shows so-defined XML elements for title in UNIMARC bibliographic record.

```
<f200>
   <ind1>1</ind1>
   <sfa>The photosynthetic bacterial reaction center</sfa>
   <sfe>structure and dynamics</sfe>
   <sfe>[proceedings of NATO Advanced Research Workshop on the Structure of the Photosynthetic
Bacterial Reaction
      Centre...held September 20-25, 1987, Cadarache, France]</sfe>
  <sff>edited by Jacques Breton and André Verméglio</sff>
</f200>
```

Example 1. Example of title in BISIS XML format

XML schema definition of so-defined XML bibliographic records and XSLT specification for transformations of these records may be used in modeling concepts of systematized verifications of UNIMARC bibliographic records. A separate definition of XML schema for each type of library material enables checking concepts of singular verifications (except lap-year verification and verification of ISBN and ISSN numbers by modulus 11) and checking concepts of cross verifications from group I and II (Table 1). However, XML schema cannot define dependences of XML elements, such as that the year of publication in `sfd` element, corresponding to subfield `100$d`, must be grater or equal to the year of publication in `sfc` element, corresponding to subfield `100$c`. For checking such constraints in other verifications (lap-year verification, verification of ISBN and ISSN numbers by modulus 11 and cross verifications from group III and IV, Table 1) the XSLT and XPath specifications may be used.

3.1. Modeling with XML Schema

Example 2 shows a part of schema definition for XML bibliographic records of monograph publications. Definitions of XML schema for XML bibliographic records of other library material types can be specified in a similar way.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:include schemaLocation="CommonTypes.xsd"/>
  <xsd:include schemaLocation="IndicatorTypes.xsd"/>
  <xsd:include schemaLocation="InternalCodes.xsd"/>
  <xsd:include schemaLocation="ExternalCodes.xsd"/>
  <xsd:include schemaLocation="Controls.xsd"/>
  <xsd:include schemaLocation="MonographsBlock9.xsd"/>
  <xsd:element name="record">
    <xsd:complexType>
      <xsd:sequence>
        ...
        <xsd:element name="f010" minOccurs="0" maxOccurs="unbounded">
          <xsd:sequence>
            <xsd:element name="sfa" type="control3Type" minOccurs="0"/>
            <xsd:element name="sfb" type="xsd:string" minOccurs="0"/>
            <xsd:element name="sfd" type="xsd:string" minOccurs="0"/>
            <xsd:element name="sfz" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
          </xsd:sequence>
        </xsd:element>
        ...
        <xsd:element name="f101" type="f101Type"/>
        <xsd:element name="f102" minOccurs="0">
          <xsd:sequence>
            <xsd:element name="sfa" type="countryCode" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="sfb" type="sf102bType" minOccurs="0 maxOccurs="unbounded"/>
          </xsd:sequence>
      </xsd:element>
      ...
      <xsd:element name="f500" minOccurs="0" maxOccurs="unbounded">
        <xsd:sequence>
          <xsd:element name="ind1" type="indType01"/>
          <xsd:element name="ind2" type="xsd:byte" fixed="0"/>
          <xsd:element name="sfa" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
          <xsd:element name="sfb" type="xsd:string" minOccurs="0"maxOccurs="unbounded"/>
          <xsd:element name="sfh" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
          <xsd:element name="sfi" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
          <xsd:element name="sfl" type="xsd:string" minOccurs="0"/>
          <xsd:element name="sfm" type="languageCode" minOccurs="0"/>
        </xsd:sequence>
      </xsd:element>
      ...
      <xsd:group ref="monographsblock9"/>
    </xsd:sequence>
  </xsd:complexType>
  ...
 </xsd:element>
</xsd:schema>
```

Example 2. Part of the schema definition for records of monograph publications

XML schema is divided among several schema documents with `xsd:include` element:

- `CommonTypes.xsd` – contains definitions of common types used in declarations of XML elements corresponding to fields that can appear in records of different library material types (for example, `f101Type` for field `101`) or that have the same structure (for example, some fields in block 3 contain only subfield `$a`),
- `IndicatorTypes.xsd` – contains definitions of types used in declarations of XML elements corresponding to indicators (for example, `indType01` for indicators that may have value `0` or `1`),
- `InternalCodes.xsd` – contains definitions of types used in declarations of XML elements corresponding to subfields that have values defined by internal lists of codes (for example, `sf102bType` for values in subfield `102$b`),
- `ExternalCodes.xsd` – contains definitions of types used in declarations of XML elements corresponding to subfields that have values defined by external lists of codes (for example, `languageCode` for values in subfields that contain languages codes),
- `Controls.xsd` – contains definitions of types used in declarations of XML elements corresponding to subfields the contents of which have to be

additionally checked (for example, `control3Type` for verification of ISBN numbers, according to the country of publication, is defined on the basis of the already defined simple type [15]), and

- `MonographsBlock9.xsd` – contains definition of group `monographsblock9` that contains declarations of XML elements corresponding to fields from block 9 for national use.

These definitions of XML element types are separated in a few documents with the purpose of easier maintenance and improved readability. Namely, different libraries may have different code lists, especially internal code lists, and for some libraries certain verifications may be more important than others. Differences appear primarily in fields from block 9 that contain local data for a particular library, such as institution location designator, acquisition status designator, acquisition price, etc. In library practice code lists sometimes have to be changed and then values in definitions of corresponding XML elements have to be changed, too. With such an organization of schema documents, it is easy to change particular schema document for these specifics.

3.2. Modeling with XSLT

Constraints of systemized verifications, that cannot be modeled with XML schema, can be modeled with syntax of XSLT and XPath languages that check values in some XML elements concerning the existence or values of other XML elements in the same XML document or in the XML document that contains upper-level XML bibliographic records. In case of incorrect values, the text with identification record number (`ID`), level of error importance (`FATAL`, `WARNING`, `INFORMATION`) and error description will be outputting as in the following example:

```
ID=123456
FATAL - Subfields 100bcd: for reproductions (100b='e') year of publication 2 must be greater or equal to
the year of publication 1.
```

The XSLT stylesheet for XML bibliographic record is defined with `xsl:template` element that transforms the root element `record` into the result tree that contains text nodes with descriptions of errors in an XML bibliographic record. Example 3 shows a part of this template definition.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="txt"/>
<xsl:template match="record">
```

```
  <xsl:variable name="ID" select="f000/sfx"/>
  <xsl:text>&#10;&#10;ID=</xsl:text>
  <xsl:value-of select="$ID"/>
  <xsl:variable name="sf100b" select="f100/sfb"/>
  <xsl:variable name="f436ind2" select="f436[1]/ind2"/>
  <xsl:variable name="f700" select="f700"/>
  <xsl:variable name="f710" select="f710"/>
  ...
  <xsl:if test="string-length(./f071) > 0 and not($sf001b='c' or $sf001b='i' or $sf001b='j')">
    <xsl:text>&#10;WARNING - Field 071: field 071 can be used for printed music scores and sound
recordings
     (001b='c','i','j').</xsl:text>
  </xsl:if>
  ...
  <xsl:choose>
    <xsl:when test="$sf100b='e' and not(number($sf100c) &lt; number($sf100d))">
       <xsl:text>&#10;FATAL - Subfields 100bcd: for reproductions (100b='e') year of publication 2 must
       be greater or equal to
       the year of publication 1. </xsl:text>
    </xsl:when>
    ...
  </xsl:choose>
  ...
  <xsl:for-each select="f101/sfa">
    <xsl:variable name="sf101a" select="."/>
    <xsl:for-each select="../sfb|../sfc">
     <xsl:if test="$sf101a=.">
       <xsl:text>&#10;WARNING - Subfields 101abc: languages codes in 101bc cannot be equal to the
language code in
        101a.</xsl:text>
     </xsl:if>
    </xsl:for-each>
  </xsl:for-each>
  ...
  <xsl:for-each select="f436[position()>1]">
    <xsl:if test="$f436ind2 != ./ind2">
     <xsl:text>&#10;WARNING - Field 436: all 436 fields mast have the same value of second
indicator.</xsl:text>
     </xsl:if>
  </xsl:for-each>
  ...
  <xsl:if test="string-length($f700) > 0 and string-length($f710) > 0">
    <xsl:text>&#10;FATAL - Field 700,710: 700 and 710 fields cannot appear in the same time.</xsl:text>
  </xsl:if>
...
</xsl:stylesheet>
```

Example 3. Part of XSLT template for root XML element `record`

Elements `xsl:if` and `xsl:when` define conditions on which text nodes, generated by nested `xsl:text` elements that contain descriptions of errors, will be outputting into a result tree (in case of incorrect contents in particular XML elements). Repeatable fields and subfields are modeled with repeatable XML elements. Sometimes, all of these repeatable XML elements must be checked. This can be defined with `xsl:for-each` element that contains a template, which is instantiated for each node, corresponding to repeatable XML elements. The verification of ISBN and ISSN numbers by modulus 11 is modeled with specific templates based on the template [16] that has already been defined.

Error messages are defined in `xsl:text` elements and every institution may conform to their need of use. This is one of the advantages of modeling verifications with XSLT specification, because error messages are not a part of some program code and so they are independent from implementation. Similar, users may delete or add conditions on which error messages will be output. That enables users to decide by themselves which verifications of bibliographic records will be performed.

4. Quality Control System of XML Bibliographic Records

The quality control system of XML bibliographic records in UNIMARC format, is determined on the basis of the defined XML schema and XML specification. This system captures all of the systemized verifications, the concepts of which are modeled with these XML specifications. The result of checking the quality of an XML bibliographic record in this system is the information concerning the validity of this record according to modeled verifications. In the case of a non-validated record, the list of error messages is created. This list presents important record quality information, because it contains warnings of missing and inadequate data or suggestions for cataloguing of dependent bibliographic data. In addition to that, the list contains error importance levels (information, warning, fatal).

This system enables quality control of bibliographic records prior to saving them into a library system. Thus, cataloguers may get information whether the quality of some record is good enough for storing the record in their library system. The list of error messages helps correct deviations in records to increase records quality. This is very important for library systems that support cooperative cataloguing, because it prevents receiving bad-quality records from the central catalog of bibliographic data.

The need of quality control of bibliographic records becomes significant in record exchange with other library systems. As library systems support different bibliographic formats, after record exchange, records usually have to be converted into the bibliographic format implemented in the local system. Thus, it may cause the loose of bibliographic data and decrease bibliographic record quality level. Therefore, it is recommended to check record quality and correct errors if possible, prior to saving records into the library system. It may be performed with this quality control system of bibliographic records. Even if record exchange is performed between library systems that implement the same bibliographic format, the quality of exchanged records may not always satisfy the quality of records in the local library system, because sometimes the same library material is catalogued in different ways, depending on library practices accepted in various library systems.

Different conversions of bibliographic databases, which are not so rare, may cause decrease in bibliographic record quality level. For example, the list of codes or accepted library practice for library material processing may sometimes change. This may result in deleting or adding of fields or subfields in almost all records of a bibliographic database. In such cases a great amount of bibliographic records in a local or central database (if library system supports cooperative cataloguing) must be corrected. Then, usually some conversion program has to be created that will, to a certain extent, change the contents in bibliographic records. Conversion rules are not always simple and may result in non-accordance of converted bibliographic data. Because of this, it is recommended to concurrently check quality of all bibliographic records in the bibliographic database, for example with this quality control system of bibliographic records. On the basis of the results of the control it might be possible to determine the quality of bibliographic records in the database and to decide if additional manual or program corrections of records are needed or not.

Bibliographic records may be compared in a view of their quality. For example, if bibliographic records, corresponding to the same bibliographic unit, may be received from different library systems, it is recommended to perform quality control of all these records prior to saving the best record into the bibliographic database. Quality control of bibliographic records finds deviations in record structure and contents, in view of syntax/semantic rules of certain bibliographic formats and of particular cataloguing rules. Thus, the best quality bibliographic record may be determined as it has the smallest number of deviations among all the records.

The quality control system of bibliographic records may also be used in a library system for creating different analyses of bibliographic data or for creating different statistics and reports, as well as for education and training of cataloguers or for checking their work.

4.1. System Implementation

Implementation of quality control system of XML bibliographic records parses XML schema and XSLT specification and saves them into the system. Each XML bibliographic record from input XML document is validated and transformed according to saved XML specifications. Validation messages and the text that is the result of XML record transformation are then saved in an output file. Afterwards, this enables error processing in XML records. GUI (*Graphical User Interface*) is simple and is shown in Figure 1.

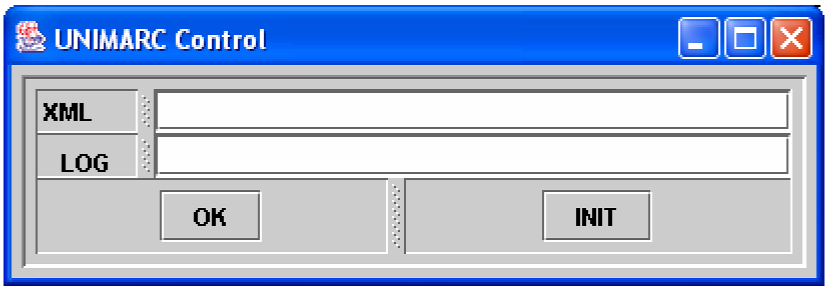


Figure 1. GUI for quality control system of XML bibliographic records

The text field labeled with *XML* contains the name of the input file that holds XML bibliographic records. The text field labeled with *LOG* contains the name of the output file that will include error messages in records after quality control is performed. *OK* button submits control of all input XML bibliographic records and writes errors into the output file. The text field labeled with *XML* may contain the names of files that hold XML schema and XSLT specification defined for some other bibliographic format. In this case *INIT* button submits initialization of new quality control system for bibliographic records according to this bibliographic format. Further controls of XML bibliographic records will be performed according to the new XML specifications.

The system is implemented in the Java programming environment, with the following ready tools for working with XML documents: for validation - MSV (*Sun Multi-Schema XML Validator*) [17], for parsing - Xerces [18], and for transformation - Xalan [19]. They have the following characteristics:

- they are in the open source domain,
- they implement standard interfaces: JARV (*Java API for RELAX Verifier*) [20] for validation, TrAX (*Transformation API for XML*) [21] for

transformation and JAXP (*Java API for XML Processing 1.2*) [22] for parsing XML documents,

- they support specifications of XML Schema Language, XSLT, XPath, DOM [23] and SAX [24],
- MSV enables schema caching that makes validation of XML documents quicker, as schema is parsing into the memory only once,
- MSV is fail-fast and reports all validation errors at once, unlike for example, XMLSpy editor [25] that reports only the first validation error,
- Xerces enables caching of XSLT specification,
- MSV and Xerces are thread-safe for using of saved XML schema and XSLT specification, which is important in developing Web Services for quality control of XML bibliographic records.

Components for XML validation, parsing of XML documents and XSLT transformations may be replaced with some other components that implement particular interfaces. Correlation of these components with components developed within system implementation is represented in Figure 2.

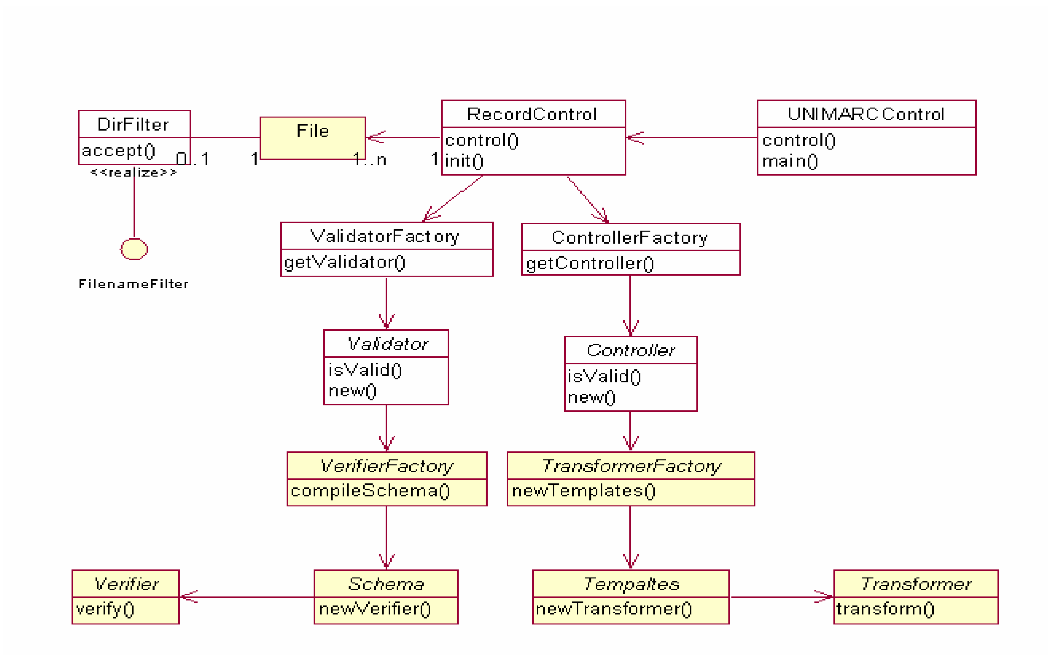


Figure 2. Quality control system of XML bibliographic records

The most important classes used in application are:

- classes from package *org.iso_relax.verifier* that is implemented by MSV:
  - *VerifierFactory* to create an instance of MSV implementation of XML validator,
  - *Schema* to store compiled schema (it is thread-safe, so even if you have more than one thread, you only need one instance of *Schema*; you can share that one instance with as many threads as required),
  - *Verifier* to perform the actual validation and check documents against a particular schema (it is not thread-safe, so typically you want to create one instance per one validation or one thread),
- classes from packages *javax.xml.transform*, *javax.xml.transform.stream*, *javax.xml.transform.dom* and *javax.xml.transform.sax* that are implemented by Xalan:
  - *TransformerFactory* to create an instance of Xalan implementation of XSLT processor,
  - *Templates* for runtime representation of processed transformation instructions (it is thread-safe for a given instance over multiple threads running concurrently, and may be used multiple times in a given session),
  - *Transformer* to transform a source tree into a result tree (it is not thread-safe),
- classes developed within application:
  - *UNIMARCControl* - GUI component,
  - *Validator* - interface for validation of XML bibliographic records,
  - *ValidatorFactory* to create an instance of implementation of *Validator* interface that is using MSV implementation of XML validator,
  - *Controller* - interface for XML bibliographic record verifications according to XSLT specification,
  - *ControllerFactory* to create an instance of implementation of *Controller* interface that is using Xalan implementation of XSLT processor,
  - *RecordControl* to connect GUI with the implementation of *Validator* and *Controller* interfaces,
  - *DirFilter* - implements *java.io.FilenameFilter* interface for filtering file names with XML bibliographic records.

The advantage of so-defined quality control system is independence from bibliographic format. Thus, the implemented system may be used for bibliographic record control, according to some other bibliographic format for which equivalent XML schema and XSLT specification are defined. Thus, it is not necessary to develop a new application, because the existent program may be used with this new XML schema and XSLT specifications. Thereby, it is not necessary to restart the running application.

The implemented system may be extended with a more sophisticated GUI or with a different organization of error messages. Additional library system functions may be developed on the basis of this system, for example an XML editor of bibliographic records that does not require any knowledge of XML on the part of users, or certain Web Services for bibliographic record control. For example, for creating such an editor, GUI must be replaced with some implementation that will enable entering data into the library system and outputting error messages in input data in a user-friendly way. The other components of quality control system of bibliographic records may remain the same.

5. Conclusion

For determination of bibliographic record quality some verification of their structure and contents has to be accomplished. These verifications may be divided into singular and cross verifications, in a view of the number of fields, subfields and indicators on which they are performed. Singular verifications check structure or contents of one single field, subfield or indicator, independently from the appearance, structure and contents of other fields, subfields and indicators. Cross verifications check structures or contents of some semantically connected fields, subfields or indicators. For UNIMARC bibliographic records in the BISIS library software system, the XML format is defined by XML schema that enables modeling of many introduced verifications. Modeling verifications, which cannot be modeled with XML schema, are performed with XSLT specification. The quality control system of XML bibliographic records is defined on the basis of XML schema and XML specification. It is implemented in the Java programming environment with ready XML tools from the open source domain. The implemented system is independent from the bibliographic format. Thus, this implementation may be incorporated into some other library information system and in the library system network.

References

[1] Bray T., Paoli J., Sperberg-McQueen C. M., Maler E, François Y. Extensible Markup Language (XML) 1.0 (Third Edition). *W3C Recommendation* 2004. http://www.w3.org/TR/2004/REC-xml-20040204/ [4 February 2004].

[2] Fallside D. C. XML Schema Part 0: Primer. *W3C Recommendation* 2001. http://www.w3.org/TR/xmlschema-0/ [2 May 2001].

[3] Tomphson H. S., Beech D., Maloney M. Mendelsohn N. XML Schema Part 1: Structures. *W3C Recommendation* 2001. http://www.w3.org/TR/xmlschema-1/ [2 May 2001].

[4] Biron P. V., Malhotra A. XML Schema Part 2: Datatyps. *W3C Recommendation* 2001. http://www.w3.org/TR/xmlschema-2/ [2 May 2001].

[5] Clark J, editor. XSL Transformations (XSLT), Version 1.0. *W3C Recommendation* 1999. http://www.w3c.org/TR/xslt [16 November 1999].

[6] Clark J., DeRose S., editors. XML Path Language. *W3C Recommendation* 1999. http://www.w3.org/TR/xpath [16 November 1999].

[7] UNIMARC Manual: bibliographic.format / International Federation of Library Associations and Institutions. IFLA Universal Bibliographic Control and International MARC Programme, New Providence, London, 1994.

[8] Library information system BISIS. http://libsrv.im.ns.ac.yu [2004].

[9] Surla D., Konjović Z., et al. Instruction Manual for Library Software System BISIS version 3, Group for Information Technologies, Novi Sad, 2003. (in Serbian)

[10] IFLA. Family of ISBDs : Publication list. International Federation of Library Associations and Institutions. http://www.ifla.org/VI/3/nd1/isbdlist.htm [September 2003].

[11] Delsey T. The Logical Structure of the Anglo-American Cataloguing Rules – Part I. *The Joint Steering Committee for Revision of AACR* 1998. http://webdoc.gwdg.de/ebook/aw/1999/jsc/aacr.pdf [August 1998].

[12] Budimir G. Quality Control of XML Bibliographic Records, master theses. Faculty of Science, Department of mathematics and informatics, Novi Sad, 2004.

[13] Budimir G. MARC records and XML. INFOTHECA, Journal of Informatics and Librarianship, 2004; 4(1).

[14] Zeremski M. Modeling of UNIMARC Format in XML Technology, master theses. Faculty of Science, Department of mathematics and informatics, Novi Sad, 2002. http://diglib.ns.ac.yu/ndltd/docs/set1/ndltd64/ZeremskiMMagistarskaTeza.pdf [2002].

[15] Costello R., Sperberg S. XML Schema simpleType Definition of an ISBN. *Xfront* 2004. http://www.xfront.com/isbn.xsd [2004].

[16] Costello R., Sperberg S. XSLT-validation of an ISBN. *Xfront*. http://www.xfront.com/isbn.xsl [2004].

[17] Kawaguchi K. Sun Multi-Schema XML Validator. *Sun Microsystems, Inc*. 1994-2004. http://www.sun.com/software/xml/developers/multischema/ [April 2003].

[18] Xerces. *The Apache Software Foundation*. http://xml.apache.org/xerces-j/ [1999, 2000].

[19] Xalan. *The Apache Software Foundation*. http://xml.apache.org/xalan-j/ [2003].

[20] Kawaguchi K. JARV User's Guide. http://iso-relax.sourceforge.net/JARV/JARV.html [15 Jan 2003].

[21] Java API for XML Processing (JAXP). *Sun Microsystems, Inc*. http://java.sun.com/xml/jaxp/ [1994-2004].

[22] Transformation API for XML. *The Apache Software Foundation*. http://xml.apache.org/xalan-j/trax.html [2003].

[23] Le Hors A., Le Hégaret P., Wood L., Nicol G., Robie J., Champion M., Byrne S., editors. Document Object Model (DOM) Level 2 Core Specification, Version 1.0. *W3C Recommendation*, 2000. http://www.w3.org/TR/DOM-Level-2-Core/ [13 November 2000].

[24] Simple API for XML *(SAX)*. http://www.saxproject.org/ [2004].

[25] XMLSpy. *Altova*. http://www.xmlspy.com [2004].

What are the Perspectives of Academic Libraries in the Information Era?

Stela Filipi-Matutinović
Acting Director, University Library "Svetozar Marković", Belgrade
stela@unilib.bg.ac.yu

Abstract. Academic libraries have some kind of hybrid position, belonging to higher education and to library world at the same time. Evolution of higher education, changing habits in scientific communication, life-long learning processes and the spreading of distance learning push the academic libraries to intensify their work and to shift from collection centered policy to user-centered policy. Digitization of collection, user education, 24 hours reference services, intensive cooperation between academic and other libraries in the country and abroad are some main issues. Discussions about the future of academic and other libraries on 69th IFLA conference showed that no matter from which part of the world they come, librarians believe that libraries are here to stay, not as museums of books but as focal points for people seeking information.

1. Age of Learning

Our time is usually described as the Age of Information. It is the fact that in the history of mankind never was such a huge amount of information about all topics on disposal to everybody who is interested. The assumption that accumulated information leads to knowledge, however, is not quite right. Knowledge requires context and human interpretation and understanding. Not the number but the quality of information is the critical factor in acquiring knowledge.

What is most important for the development of mankind is the human ability to learn. We all know that many facts we learn during the formal education process or during professional development quickly become obsolete, and the period for which some knowledge is valid is getting shorter every year. So everybody has to learn new facts, new procedures, to adapt to new technology all life long. Apart from not being easy, this process is expensive – takes a lot of time, energy and resources. If we suppose that people are motivated to learn, because of in-bourn curiosity or hoping to get better position in the work process, or even to keep the position they already have, there is still the problem of resources.

To make the learning process available to everybody, there should be an institution open to everybody, with up-to-date resources and staff ready to help people to find their way in the enormous quantity of information and to get what they need. The institution should have stabile funding and qualified staff who could choose how to spend the existing financial resources to provide information resources which would cover most of the requests.

In the history the only institutions of that profile were libraries, founded and supported by states, local communities or different educational or research institutions. During their history libraries evolved from institutions belonging to higher classes to public institutions on disposal to much broader range of users.

2. Evolution of Higher Education

Higher education is in the process of big changes all over the world. In the last century, universities evolved from institutions which were accesible to small part of population to institutions which must transform themselves to some kind of industry of education. In developed countries the number of young people involved in higher education grew from less then 10% of population to more then a third of population, and there are expectations that the percent is going to increase. That enourmous pressure on those institutions led to a lot of changes in the way the education is performed. There is a shift from learning through lectures given by professors in the class to guided, but informal, learning where much more is left to students to learn by themselves, through working and discussion with professors and colleagues on the subjects they choose.

There is much more flexibility and much more independence in creating programs during the study. The modern university education has to teach students how to find and use newest information for practical purposes or for scientific research. It is not possible any more to teach students a definite corpus of knowledge which will be sufficient for their professional career.

3. Communication System in Science

Science can not exist without its environment, and it is dependent on the inflow of resources and information, homeostatic characteristics like tradition, education system etc, development of science depending on its inherent characteristics and environmental limitations. There is a analogy by Thomas Blackburn, which compares science as a social system and ecosystem. Scientists produce, structure and exchange information and ecosystems produce, structure and exchange biomass. Science on the universities is dependent both on the research and communication activities. [1]

As a part of communication system in higher education, universities have always organised conferences, lectures, evaluated doctoral theses, provided editorial boards and referees for their publications and issued reports. Libraries and university presses were ju elements in a highly fragmented scholarly communication process. Development of information and communication technology changed the ways of communication completely, and the IT revolution started at the universities.

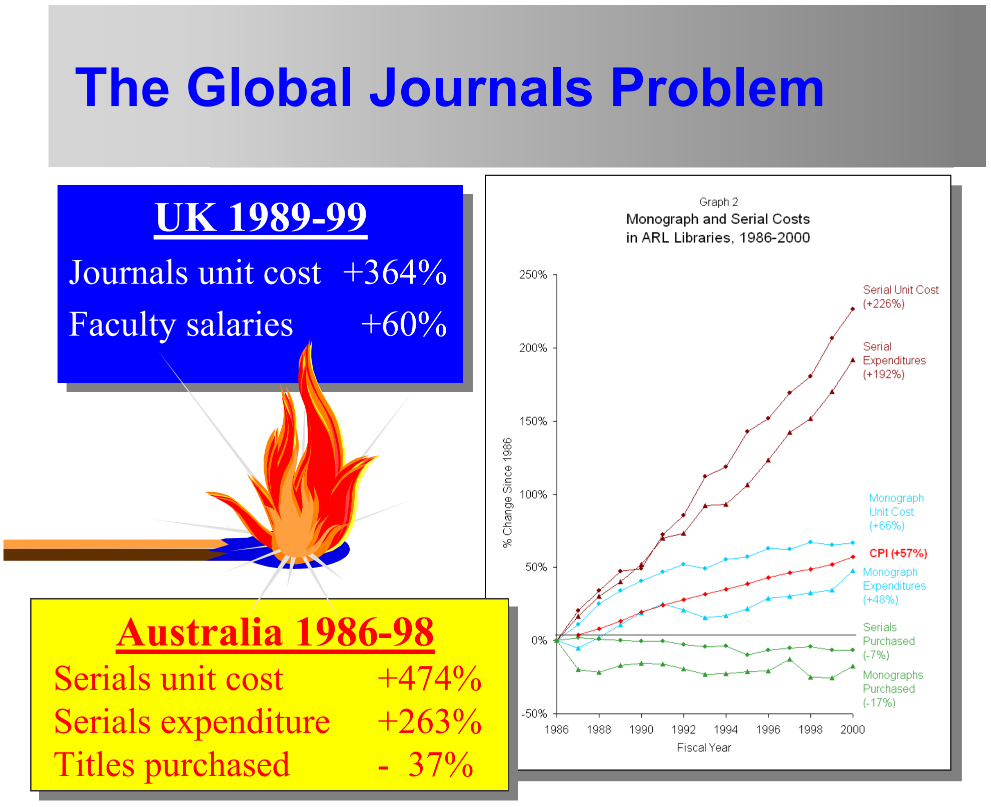
Communication system in science is nowadays becoming a limiting factor for its development, in spite of all the new communication technology. The main communication chanals in science still are peer-revied scientific journals. There are about 24.000 journal titles, and every year their prices are increasing. In the age of print the prices were lower and more academic libraries were able to buy bigger percent of all those journals and give acces to them to users, mostly scientists and students.

David Prosser, SPARC (Scholarly Publishing & Academic Resources Coalition) Europe Director, in his presentation: "The Next Information Revolution – Can Institutional Repositories and Open Access Transform Scholarly Communications?" gave the data, showing where is the problem which all the scientist and libraries in the world have to cope with, even in rich countries like UK and Australia (see Fig. 1). The number of journals is increasing, their prices for paper or electronic versions are increasing, and library budgets of universities are not increasing, or are even decreasing in a lot of countries.

In the late nineties and in the new millennium World Wide Web is getting more and more important position as a communication channel, even in science. The role of libraries is to give the free access to WWW for their users and educate them how to find and evaluate properly the information available on the web.

Institutional or subject repositories with open access, which are also very important information resources, offer the chance for research and library communities to take back control of scholarly communication. They should collect digital content generated by institutional or professional community: preprints and working papers, published articles, teaching materials, student theses, data-sets, etc. and give global online free access. Open Access widen dissemination, accelerate research, enrich education, share learning among rich & poor nations, enhance return on taxpayer investment in research. Academic libraries are the right places to maintain repositories of their universities and to promote principles of open access.

On the Online conference 2002 in London nobody mentioned Open Access movement and open access journals as important issue in building library collections, but on Online conference 2003 some speakers mentioned that it is possible that in the future more and more journals will move to open access, and discussed about the consequences for library collection building policy and for users [2].



The Global Journals Problem

UK 1989-99

Journals unit cost	+364%
Faculty salaries	+60%

Australia 1986-98

Serials unit cost	+474%
Serials expenditure	+263%
Titles purchased	- 37%

Figure 1. The global journals problem

4. Distance Learning

Distance education programs are now standard in most colleges and universities in the developed countries. Academic libraries have begun to build different online services and programs for remote users. The ACRL *Guidelines for Distance Learning Library Services* (ACRL, 2000) state that, "Members of the distance learning community are entitled to library services and resources equivalent to those provided for students and faculty in traditional campus settings". On the ranking list made by students at the University of Iowa [3], the following services were ranked highest: Web and/or e-mail reference 71.3 percent, remote access to full-text databases 65.1 percent, home delivery of books and articles 60.7 percent. Guides to doing library research were ranked tenth with only

30.9 percent. According to some other investigations, [4] the key library services to remote users are:

- Library resources available in full-text online, including electronic journals, full-text databases, digitised collections, e-books, etc. that are accessible remotely.
- Effective and expedient document delivery services if students are to access resources that are not available in full-text online.
- The necessity for "point of need" assistance - in any of the forms: virtual reference, online tutorials, e-mail or telephone communication with librarians, and other instructive resources that enhance students' information literacy experiences.
- The need for off-campus students to have access to a local library to complement the resources and services available to them online. This is reflective of the fact that the library has not lost its value as a physical space. i.e. as a place for quiet study, or a place to go and photocopy material, or even as a place for group discussion of an assignment or project.

Academic libraries in less developed countries have to prepare for providing service to distant users, because sooner or later they will have to serve them.

5. Digitization of Collection

Digitization and growth of scientific materials produced in electronic form provided the environmental conditions for changes. While the universities and their libraries expected that digitization would make scholarly communication cheaper, it is not the case. When librarians wanted to use the Internet to disseminate knowledge, they discovered that the access to electronic resources is often restricted by monopolies, and that the electronic journals once free, have prices that are growing every year. Some of the librarians discussing at Online conference 2003 about how electronic journals changed our life talked about “coccaine marketing” of electronic journals – they are free at first, and when people get used to them, charging for use is involved. When universities started to stimulate prior publishing of articles on pre-print servers, they found out that this could inhibit formal publication in a prestigious journal. The problem is more complicated because of widely used method of evaluation of scientists and institutions according to Impact Factor of journal they publish in and the number of citations they received according to citation indexes produced by ISI, now Thompson. May be the fact that open access journals get more citations than others will help in changing the situation. This is going to be a long battle between universities as main producers and consumers of electronic information and big publishers which managed to became monopolists.

Instead of moving into a freer world, we entered a world of licensing contracts with complicated legislation and new terms like authorized user, permitted user, registered user, walk-in-user, selected user, perpetual access, IP-domain and site, fair use, etc. All these terms have one thing in common: the risk of not meeting the requirements is always laid on the side of users. All this occurred at the same time with a great lowering of production costs on the publishers' side by the new digital technology. The costs are even further reduced by enforced bulk purchase via mandatory mega licenses that come in place of the annual fine-tuning of individual library collections. An extra side effect of the monopolistic control that publishers have is that the usage profile per IP-address is monitored, that is per machine, or per password, which means per user. In the corporate arena of knowledge-based competition, this is invaluable strategic information. Companies hardly seem to be aware of that.

Most university and other research libraries have limited financial resources, have to make cancellation for journals, buy less and less books, and planning "massive digitization projects" to enable all print materials to be accessed over the Internet is unrealistic. The current cost to convert books and collections is estimated at $1 per page. Budget realities eliminate the massive digitization of library materials in academic libraries for the foreseeable future. But what they have and can do is to collect, organize and preserve the scientific and educational materials already produced in digital form and to fight for rights of their users to have access to the electronic resources with as little restrictions as possible.

The academic libraries should orient on building repositories for the university materials that are produced in digital form and connect themselves closely with university publishers and computer centers. If they do not react quickly and build strong connections with their universities and their university computer centres, their future can be pretty uncertain.

6. User Education

Information literacy is an issue for every college and university [5]. Librarians should occupy a unique position in attempts to define and achieve information literacy of students. The goal of producing students knowledgeable in their disciplines and capable of adjusting and advancing in college and life after college is universal to all institutions of higher learning. Academic librarians are involved in user education, but that brought new problems. They were never really considered legitimate faculty members, not even when its members had that official status. Yet, it was library associations, library affiliated institutions, and their advocates who brought information literacy to the attention of the society. Librarians were the most prepared by their training, professional inclination, and commitment, to initiate the processes, supply the expertise, and define the framework within which those goals could be accomplished. They also appeared to be the ones most committed to that goal.

The academic library was thus the natural place for designing solutions toward information literacy. But with practices that leaves majority of students without instruction, the deficiency of existing systems of library instruction could not be denied. Any far-reaching solution had to be comprehensive and diversified. An integrated approach with a programmatic arrangement that ensures a plan for engaging the entire student population could ensure that comprehensiveness and diversification. What distinguishes information literacy as a more ambitious instructional engagement than conventional bibliographic instruction is understanding the need to go beyond teaching mainly retrieval skills, but to learn about and understand total research environment in the course of finding and using information for acquiering knowledge. Conventional bibliographic instruction devoted a great deal of time and energy to teaching the way of finding information, not the evaluations of the retrieved information. Multiple studies have confirmed what many of librarians in the developed countries have observed: students rely on the Internet as the primary source of information for coursework, neglecting library databases and print resources, and about 20 percent of them never make a judgment about the quality of the information that they obtain from the Internet or other sources.

Committing to teaching students to find, evaluate, and use information and knowledge in a dynamically evolving universe required an integration of the librarian's qualifications into the requirements of the curriculum. That is the case in United Kingdom, where key skills which include a lot of practical knowledge, necessary for studing and future work, are part of the curriculum at all high education institutions, and librarians are involved in preparing programs and teaching key skills concerning information literacy. [6]

The Commission on Higher Education of the Middle States Association of Colleges and Schools in USA expressed the view that librarians have to play an important role in the information literacy endeavors of higher education. The academic library will have to be elevated to the status of an academic teaching department, a future that some find both necessary and pragmatically expedient. Regardless of whether there is such a department or not, the library should offer an independent credit course in information literacy, one that offers in-depth engagement with issues inherent in and skills attendant to information literacy. Ideally, this course will become part of the general education curriculum, and a prerequisite for graduation. Such a program should reach as broad student population as possible, at the early stages of their studies, because information literacy skills are not only vital to effective research and improved success of the educated professional, but also fundamental to the creation of the competent student. The American Library Association starts from that position when insisting on the need in education for a new learning model that recognizes that "teaching facts is a poor substitute for teaching people how to learn."

In other countries all over the world there are different models of user education in academic libraries, and the trend is to include some kind of user education is spreading. Plain bibliographic instruction is not enough for modern times. This task is going to be very important for future academic librarians, but it also means that they will have to learn a lot and always be one step in front of the average technological knowledge of their students. That is not an easy task with such quick evolution of communication and information technology. To fulfill this task a lot of academic librarians themseves should become “life long learners”. Job of an academic librarian is getting more and more dynamic and there will be no space in modern libraries for people who are not ready to adapt to this new environment.

7. Library Cooperation

Libraries started to cooperate by exchanging publications, through interlibrary loan and corodinated acquisition long time ago, and the cooperation became more intensive after the Second World War. There were many factors influencing the cooperation, ranging from ideals of universal availability of publications to all library users in the world, through economic restrictions that forced libraries to make the system of coordinated acquisition and technological development which enabled them to start building networks and exchange records. Establishment of national and regional union catalogues and networked cataloguing through the Online Computer Library Center (OCLC) was one of the most obvious examples of cooperation in the last quarter of 20th century. Cooperation expanded rapidly and provided valuable support to libraries facing economic difficulties characterised by unsufficient budgets and increasing costs of library materials. In addition, the important fact was the increasing availability of databases on CD-ROM and on line (Medline, Inspec, Pascal, etc.) which had become accessible in libraries through document supply services. Furthermore, prices for online database searches dropped significantly during nineties. The cooperation between libraries therefore continued to grow due to increasingly broad database availability and greater flexibility for users.

With the spread of WWW commercial publishers were quick to offer electronic resources such as serial publications, electronic versions of printed periodicals or completely new electronic journals. At the beginning they were available free of charge, in order to create a demand, but fees were introduced later. Most products were in the areas of science, technology and health, so users from these fields were the first to take an interest. Among the many perspectives opened up by the advent of the Internet, online access to the full text of scientific and technical information (STI) has been the most directly relevant to university libraries. Scientific information should be taken here as referring to published - and therefore peer-reviewed - information. University libraries have become well aware of the potential of the new information media to improve their STI access services. Trend in the new millenium shows that libraries with the smaller collections of printed material began to catch up with other, bigger libraries, if they join the national or regional library consortia and get access to many more serial publications in electronic form than they ever had in printed form. The second change was a shift to multidisciplinary document searches. The union catalogues as the result of merger between the separate library catalogues and shared cataloguing systems were introduced in many countries and regions, and have been freely available for consultation through Web OPAC interfaces [7].

Academic library users need a standardized yet easy-to-access electronic content environment that can be comprehensively searched by specialized and highly focused search interfaces. "One-stop shopping" is an idea central to the aim and purpose of libraries. [8] Libraries themselves were created as a one-stop shopping tool - a centralized place where disparate sources of knowledge are brought together to be easily retrieved. Online library catalogs are also metasearch tools. The Integrated Library System has allowed libraries to bring together formerly diverse collections to be searched together. Library consortiums and Z39.50 protocol for data exchange were developed at least in part to allow metasearching of materials from many libraries.

The print indexing and abstracting tools on which today's online serials databases and full-text research tools are based were created as "one-stop shopping solutions" for journal materials, in particular, subject disciplines. Subject specific indexes such as Medline, Agricola, Inspec, Psychological Abstracts, Sociological Abstracts etc, were created to provide a kind of metasearch access to the core of scholarly literature in particular fields of study. These resources in their electronic form are still essential tools for accessing journal materials in subject disciplines. Aggregators such as Dialog changed the concept, aggregating the databases and providing metasearch capabilities that allowed for cross-file searching of most important scholarly literature from different disciplines.

Metasearching capabilities we have today are no longer sufficient. The range and volume of online resources available in libraries have expanded. User expectations for ease of searching and access have also grown beyond the capabilities of traditional search tools. Libraries have greatly increased the amount of available material, but the complicated database environment has moved libraries further away from, not closer to, the goal of “one-stop shopping”.

The number of interfaces continues to expand as new e-reference products and electronic books are added. Existing interfaces are changing rapidly as well and the problem of complexity is worsening. Library users are generally not interested which publisher or aggregator is responsible for a journal. They are primarily concerned with locating good quality material that is relevant for their research topics. Publisher-specific databases do not attempt to cover the whole literature of any particular subject area or field of study. These databases may focus on particular subjects, but they feature content from only one publisher and exclude material from competitors. Aggregator databases most often cover a wide range of subjects, rather than attempting to focus on the comprehensive coverage of any single subject and generally cannot include content provided by many publisher-specific products.

A standardized, cross-searchable, uniform e-content environment is both essential and inevitable. The success of Google originated because of the HTML standardized environment of the World Wide Web. PC software, electronic text, and commercial databases became largely standardized as they developed, and the library e-content environment needs to move in the same direction.

Although offering some real immediate advantages, today's metasearch tools cannot overcome the underlying problems of growing complexity and lack of uniformity. An open and uniform e-content environment is necessary for providing interconnected and accessible environment that libraries are seeking from metasearching.

8. The Future of Libraries

A number of misleading but popular simplifications complicates the work of librarians. Perhaps the most common of these is the mantra that “everything is on the Web.” Further, while the Web is a marvelous tool for communication and the dissemination of information, it also is chaotic, ephemeral and increasingly commercialized. Web resources cited are more likely gone than active; studies suggest that the typical Web page survives between 45 and 75 days.

Keeping anything on the Web requires ongoing costs for hardware, software, communication lines and staff. Most Web information actually is designed to sell something, promote an institution, a person or a particular point of view. There’s a great deal of information on the Web, but the searcher increasingly finds that payment is expected for the "good stuff." The hope of librarians and administrators that digital materials will be cheaper than print versions has been proven wrong.

How do these insights impact the library of tomorrow? A number of questions arise:

What resources do we provide, and in what format (print or digital)?

What, specifically, are the goals of the library?

Whom — or what — do we serve and in what priority?

If there is a dominant theme that should guide decision making, it is the issue of supporting "learning" in all its forms. Libraries are ideally suited to provide multiple support systems for people who learn in different ways. Librarians should guide learners to various resources in different media; they need to provide support for learning individually and in groups; and they must offer technological support systems and individual mentoring to help students and faculty with the variety of materials and technologies available today.

Also, the library's perspective will have to be global. Ubiquitous automatic translators can facilitate truly global information-accessing programs.

We live in a "visual world," where we get information and communicate in a wide range of media, of which print is only one. This requires new levels and dimensions of literacy. But literacy is not enough. Further capabilities — or competencies — in finding information, accessing it, evaluating the material and using it to communicate with others and solve problems is expected as well.

The impact of the information revolution is not fully recognized. The "half life" of information usefulness has shrunk from a century or a generation to perhaps no more than days or even hours in some fields, where anything in print is automatically deemed obsolete. Today the information underlying the first year of a certain technical degree can be half useless by the graduation day. Currency now prevails. Librarians must grasp the significance of this transformation. Instead of guiding learners to the authoritative book or article, it will be vital to point as well to the current "best knowledge," and that often is the domain of a research group or even an individual wrestling with the problem in a given context.

There is a lot of questions and to find relevant answers is not easy. The only certain thing is that libraries will change a lot in the future, and the changes concern both their appearance and their role.

Libraries are among the best sources of common space, that is, shared places that serve the core academic mission at colleges and universities. There are four main points concerning libraries as a space [9] :

1) *Choice*. People will choose the information medium and environment that is most pleasurable and convenient. Most library customers will avoid a confusing, uncomfortable, and unattractive library.
2) *Community*. Although there are more convenient alternatives in chat rooms, DVDs, television, takeout, and Amazon.com, people continue to go to the movies, theatre, concerts, restaurants, and bookstores, even as these have become increasingly costly. The need to be with other people and the pleasure associated with the real - rather than virtual - experience offered by these special places is a powerful draw.
3) *Communication*. Though each form of communication has its strengths, if students and faculty were asked about their choice of the most effective way to communicate and collaborate, the majority would make the following selections: face-to-face, telephone, e-mail, voicemail, pager and fax, favouring direct response over asynchronous. Face-to-face delivers more information with non-verbal signals, quicker response and effective interplay, but requires place. Although people love their devices, they want to be with other people more. Even information technology employees, whether in a university or a business, still need to sit in close proximity to each other.
4) *Expectations*. A university campus must fulfill students' subconscious desire to connect and belong, a need that can be difficult to satisfy in an increasingly mobile, global, and virtual world.

An academic library can contribute to the image of its university as an attractive place in four ways:

1) as a key component of a campus location or site;
2) with a distinctive memorable exterior image;
3) by providing high quality and sensitively designed interior environments; and
4) most recently, by establishing a convenient, responsive and actively engaging virtual presence on a campus network.

In a modern academic library there should be provided for users at least four distinct levels of activity or zones of social interaction that provide recreational, study, and working environments:

1) Highly active and engaging communal places.
2) Interactive collaborative places for individual research and group work.
3) Quieter less active places such as reading/study rooms.
4) Out of the way contemplative places for quiet reflection and deep thought.

The role of the libraries has changed greatly over the last decade. Increasingly libraries are focusing on providing access to materials rather then holding the items itself – “just in time” policy instead of “just in case” policy. Most of the libraries invested heavily in electronic resources and established an efficient interlibrary loan service. Networked electronic catalogues, access to electronic resources and established digital repositories in libraries have to be maintained, and that is not the job of librarians, but of information specialists. It is not easy for libraries even in developed countries to hire good IT specialists, because they are much better paid in other institutions. A recent arrangement in UK universities and colleges is for the library and computer services department to be under a single head, who may be either the librarian or a computer scientist. Such services are reffered to as converged services, and over the last decade these have become increasingly common in UK [10]. In other countries the cooperation between university computer centres and academic libraries is also becoming a trend.

Typical academic library mission statement today claims that thee library provides integrated library and information services in support of the University overall aims and objectives and will continue to be responsible to user demand and proactive in the development of services and use of new technology.

The idea of the library as a gateway to the future through information technology is universally accepted. Library as a portal was also the motto of the 69th Congress of International Federation of Library Associations in Berlin 2003. This is a two-way portal, linking the solid proven knowledge and information resources of the past to virtual resources of the future.

An interesting book, summarizing the history of libraries in Germany and predictions of the global future of libraries was a part of materials every participant on the Congress received [11]. Part of the book were analysis of the results of interwieving eminent professionals from library, information, publishing and education world about their visions and predictions concerning future of the libraries all over the world.

Majority of interviewed information scientists believe that libraries will only have a future if they converge with computer centers and multimediua centers to create a new infrastructure for information and communication. The main task for those institution would be knowledge management, thus the coordination of all sources of knowledge completely represented in digital form. They believe that advancing of digitization would leave libraries with printed resources as book museums. Librarians and publishers believe that the role of libraries will be transformed, but that they will continue to exist with some old and some new functions.

Expected contextual conditions were summarized by the authors as follows:

- Education as such will be the primary issue in the advancing 21st century. Education will influence the entire society – above all in the commercial and economic areas
- The acquisition of information will not be the problem in the comming century, but rather concentration on the most fundamental and on the correct information: Quality instead of quantity will be the focus more then ever before
- The Marketplace “Library” which allows people to actually meet, must be designed completely differently then the isolated, lonely work place in front of the computer screen with its view into the “global village” of the internet
- The Library of 2015 must be available as a space attached to a specific physical building and as an independent part of the cultural life of a community integrated into the social life of that community.
- The relationship of the print medium and the digitally stored medium will come to rest at 50 : 50 by the end of 2015; the non-book proportion will then rise only insignificantly in the following decades.

Majority of the answers about the existence of library buildings was that the libraries will still have buildings. Information is better communicated in person to person relationship than in the machine to person combination. Users are looking for advice and training in the personal communication. It is interesting to look at the list of primary functions for future libraries, in order of frequency of occurence of answers from interviewed experts:

5) Media Center/Information Center
1) Document Center
2) Data Center
3) Lending Library
4) Call Center
5) Cultural Center
6) Computer Center
7) Full Text Server
8) Place of Learning
9) Long-term archive
10) Internet Cafe
11) Book museum
12) Citizen`s information office.

Interwieved professionals formulated differently their expectations from libraries in the future, and their answers were summarized as follows:

1) Online indexing of complete collections (retroconversion), integrating catalog data into regional union catalogs, expanding the digital cataloguing of collections to small libraries. Full text digitization of large reference works.
2) Focal point for public libraries will be learning, information, advice, participation, user education ... etc.
3) Libraries will have to take on the long-term archiving of publications. This will only be possible to be carried out in very few institutions to which others will have to turn and depend upon within a network. Similarly, cataloguing and indexing will no longer be a product of individual achievement in the future, rather it must result from a secondary harmonization in the sense of customer-orientation.

Standing Committee of University Libraries and other General Research Libraries on the 69th IFLA Congress in Berlin 2003 summarized the important subject for those libraries as follows:

1) Changing roles of librarians in academic and research libraries in order to assume new roles as information providers, educators and knowledge managers in training users to be information literate and lifelong learners.
2) Evaluation and quality assurance in academic and research libraries in order to promote the development and dissemination of best practice in quality assurance and review of services provided by academic and research libraries as part of the institutional evaluation process.
3) The library's role in shaping the future of scholarly communication and publishing and dealing with the problems in providing essential information to users all over the world, but specially in less developed countries.
4) Academic and research libraries, specially in less developed countries should understand and contribute to the debate on Higher Education issues at the beginning of the new millennium, focusing on the real situation and problems of the sector in less developed countries as well as on the ways academic and research libraries can contribute to democracy and welfare.

Libraries in many countries are working hardly on the improvement of the services for final user, providing the essential means for:

- the identification and circulation of the bibliographic information resources available in the country
- the development of inter-library loan
- the standards for professional practices
- the policy statement of a co-ordinated acquisition policy.

The conclusion was that all libraries in the world, and specially academic libraries must put the users and their expectations from the library in the focus of their work if they do not want to become book museum or interent caffee.

9. Conclusion

If academic librarians do not change according to the needs of transforming university and students grown with Internet, WWW and mobile phones, they will be replaced by "knowledge managers". In such quickly changing environment, there is no time to loose.

Librarians do tend to hold endless discussions about best ways of cataloguing and classification, but in the mean time others are making new metadata standards, new taxonomies, programs for content management etc. (2, 12). The point is that libraries must remember that all the librarians are doing has no sense if the users can not see the difference between perfect and less perfect record, and decide not to wait for "perfect" union catalogues or databases and turn to search engines instead, which are not perfect but are quick.

References

[1] Blackburn Thomas R. : Information and the ecology of scholars. – Science, vol. 181, 1973, 4105, pp. 1141-1147

[2] Filipi-Matutinovic Stela, Mazic Bogoljub: Online Information 2003 Conference. – Infoteka, vol. 5, 2004, special issue, pp. 101-105

[3] Waaijers Leo: Stratum continuum of information: scholarly communication and the role of university libraries. - New Library World, vol. 103 Number 4/5 2002 pp. 165-171
4. Mutinta Moyo Lesley, Stern Cahoy Ellysa: Meeting the needs of remote library users. - Library Management, vol. 24, 2003, 6/7 pp. 281-290
5. Owusu-Ansah, Edward K. : Information Literacy and Higher Education: Placing the Academic Library in the Center of a Comprehensive Solution. - Journal of Academic Librarianship, vol. 30, 2004, Issue 1, pp 3-17

[4] Kirk Meg: Learning Resource Centre participation in key skills activities. – Infoteka, 2003, vol. 4, 1, pp. 26-32

[5] Boukacem Chérifa: Inter-library loan services and access to electronic resources in French university libraries: a marriage of reason. - Interlending & Document Supply, vol. 31, 4, 2003, pp. 218-227

[6] Webster Peter: Metasearching in an Academic Environment. Online; Mar/Apr2004, vol. 28, Issue 2, pp 20-24

[7] Rizzo Joseph C: Finding your place in the information age library. - New Library World, vol. 103, 2002, 11/12 pp. 457-466

[8] Marsterson William: Organization of libraries in UK. – Infoteka, vol. 5, 2004, special issue, pp. 8-16

[9] Seefeldt Juergen, Syre Ludger: Portals to the past and to the future : Libraries in Germany. – Hildesheim, Yurich, New York : Georg Olms Verlag, 2003

[10] Hopkinson Alan: Metadata: an overview. – Infoteka, vol. 5, 2004, special issue, pp. 17-22

Future Improvements of the Virtual Library of Serbia (VLS) in an Academic Environment – Some Considerations

Bogoljub Mazić
Deputy Acting Director, University Library "Svetozar Marković", Belgrade
bogi@unilib.bg.ac.yu

Abstract. Development of the library information system in some faculty and institutional libraries of the three Serbian universities is a first stage of a step-by-step project of establishing university library automation system as a part of the national library system of Serbia. The development of university library automation system is partly based on COBISS organisational model based on shared cataloguing utilities combined with utilities of access to full text databases, and other end-users shared utilities. After establishing Web based union catalogue and local library systems next phases will have to be more orientated toward regional interconnections, establishing digital library and further Web based user services. Future development should bring regional cooperation between various regional COBISS centers in the neighbouring countries. They will retrieve and download records between themselves and in that way further improve the quality of their work. An extension of the bibliographic database should follow with the possibility of implementation and maintenance of the full-text databases of doctoral and master theses presented at all Serbian universities as well as the formation of the electronic repositories of the universities scientific papers. In that way, the complete scientific output of professors and researchers of the University could be accessed in an easy way and they will be able to produce their own Web based personal bibliographies without any difficulties.

1. Introduction

Building virtual libraries of universities, regions or states is a common trend in modern librarianship. Establishment of academic networks with 24/7/365 access to Internet enabled the libraries in academic community to build virtual academic libraries and gave users all over the world the possibility to use their electronic catalogues and interlibrary loan where necessary. Three existing main university libraries in Serbia were aware of that trend and when the opportunity arouse to apply for Tempus projects in 2001, they decided to join efforts in building academic libraries network for universities in Belgrade, Niš and Kragujevac.

University library in Belgrade worked at that time with COBISS software, and university libraries in Niš and Kragujevac worked with local programmes based on CDS/ISIS.

At the beginning of the 2001 National library applied to the Open Society Institute for funding the creation of the national shared union catalogue for Serbia. It was supposed that it should be based on the existing electronic records created in COBISS software in Library of Matica srpska, National Library of Serbia, Yugoslav Bibliographic Information Institute and University Library »Svetozar Marković«. When both projects were approved for funding, Consortium of the Tempus project UMI_JEP 16059-2001 and National Library of Serbia decided to join efforts in the establishment of shared union catalogue as the core of the Virtual Library of Serbia. The shared union catalogue was promoted in February 2003, with 1,300,000 records at the beginning and 3 networked libraries. Now it has about 1,500,000 records and about 70 libraries of different size, mostly academic. More than 120 librarians have licenses for working in the shared cataloguing system. Workshops and courses for training of librarians are regularly organized and necessary professional literature is being published in print and electronic form and distributed to all libraries included in the system. Electronic information, publications and Serbian professional journals for librarianship and information science are accessible on websites of University Library »Svetozar Marković« www.unilib.bg.ac.yu and National Library of Serbia www.nbs.bg.ac.yu.

The main intention of building a system that we now call Virtual library of Serbia (VLS) was to connect all libraries with important collections and developed library services into a uniform information system. Library users are provided with efficient way of access to all kinds of information, whether those generated within the system or available somewhere on the Internet.

It was a project that intended to put together information about library collections in Serbia with information and electronic documents from databases stored on any computer in the system. VLS was built according to the model of a distributed information system, supported by computer network across the libraries.

Some possibilities already present in the VLS system but not yet fully realised will be discussed here in some detail and their importance for future improvements of VLS will be stressed.

2. Integrated Access to E-journals

VLS must become in the next phase union catalogue for foreign printed journals in the Serbian libraries in the full sense of that word.

Users are not encountered with the whole and comprehensive information with today's divide of data between journals available in library collections within VLS and those financed from Ministry of science that are in collections of these and other academic libraries in Serbia.

Bibliographic records for electronic journals can be generated with the assistance of ISSN database. In this way several thousand bibliographic records could be produced in only one day that would otherwise, with manual cataloguing, require months of work. In bibliographic record field for online access links to servers (full text database web address) where full text database of the journal should be found. There is also a note about e-journal conditions of use.

This field is being created and maintained automatically with so called "e-links" database, based on data of service provider that offers access to electronic journals. It is being held accurate according to periodical lists of included journals that every provider sends to his users.

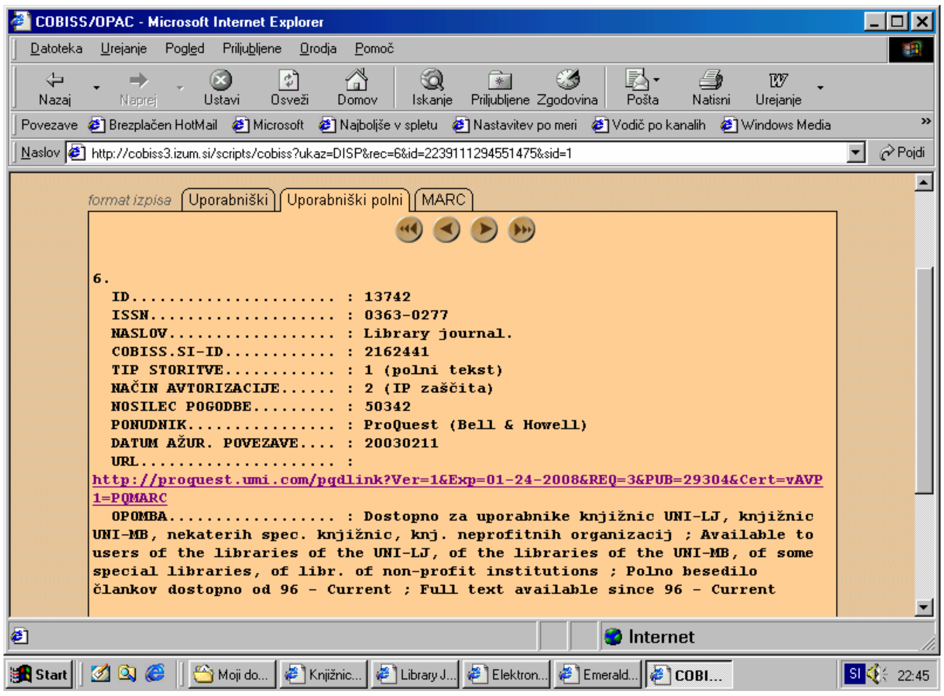
It reduces cataloguing costs and enables prompt updating that is of special importance for user. This connection stands on ISSN number as an unique identifier of each journal and is being held in both databases. Kobson portal and VBS shared database could be interconnected and users familiar with searching for journals according to title or other bibliographic data could be able to reach Kobson from bibliographic records and vice versa, Kobson users would access bibliographic records from the portals pages.

If we update shared cataloguing with a reporting system within whom one or more libraries are authorised to input data about other library's collection, the main aim of achieving complete information will be fulfilled.

Printouts of reports for data collection could be defined in the shared database itself where system controls the procedure data input by checking the entered library signatures and codes for document types.

3. Cobiss.Net

Cobiss.Net will enable exchange of bibliographic records between separate COBISS systems in four countries of former Yugoslavia. It is possible without much additional adjustments because they are all applying same cataloguing rules and systems of subject analysis.


COBISS/OPAC - Microsoft Internet Explorer
Datoteka Urejanje Pogled Priljubljene Orodja Pomoč
Nazaj Naprej Ustavi Osveži Domov Iskanje Priljubljene Zgodovina Pošta Natisni Urejanje
Povezave Brezplačen HotMail Microsoft Najboljše v spletu Nastavitev po meri Vodič po kanalih Windows Media
Naslov http://cobiss3.izum.si/scripts/cobiss?ukaz=DISP&rec=6&id=2239111294551475&sid=1
Pojdi
format izpisa Uporabniški Uporabniški polni MARC
Internet
Start Moji do... Knjižnic... Library J... Elektron... Emerald... COBI... SI 22:45

```
6.
  ID..................... : 13742
  ISSN................... : 0363-0277
  NASLOV................. : Library journal.
  COBISS.SI-ID........... : 2162441
  TIP STORITVE........... : 1 (polni tekst)
  NAČIN AVTORIZACIJE..... : 2 (IP zaščita)
  NOSILEC POGODBE........ : 50342
  PONUDNIK............... : ProQuest (Bell & Howell)
  DATUM AŽUR. POVEZAVE.... : 20030211
  URL.................... :
http://proquest.umi.com/pqdlink?Ver=1&Exp=01-24-2008&REQ=3&PUB=29304&Cert=vAVP
1=PQMARC
  OPOMBA................. : Dostopno za uporabnike knjižnic UNI-LJ, knjižnic
UNI-MB, nekaterih spec. knjižnic, knj. neprofitnih organizacij ; Available to
users of the libraries of the UNI-LJ, of the libraries of the UNI-MB, of some
special libraries, of libr. of non-profit institutions ; Polno besedilo
člankov dostopno od 96 - Current ; Full text available since 96 - Current
```

Figure 1. Connection between "e-links" database and bibliographic database servers for automated maintenance and updating of field for online access.

4. Personal Bibliographies

Union database catalogue VLS should be a source of personal bibliographies creation for university staff in Serbia and other scientific workers in the country. Systematic cataloguing of Serbian publishing output together with documents of other scientist and professors, should enable them with web version of VLS shared catalogue to produce their own personal bibliographies in one of the standard citation format.

Broadening of VLS programme to this field of scientific work is based on introduction of union code lists of scientific workers and organisations and typology of scientific documents. Entry of bibliographic records should be done by networked libraries responsible for correct entry of data for their respective scientific staff as well as for those libraries not yet in the system.

Code lists of scientists rests on unique identifier of each person and is organised according to current number in order to provide clear identification of scientific output that can be distinguished from works of other authors with same or similar names.

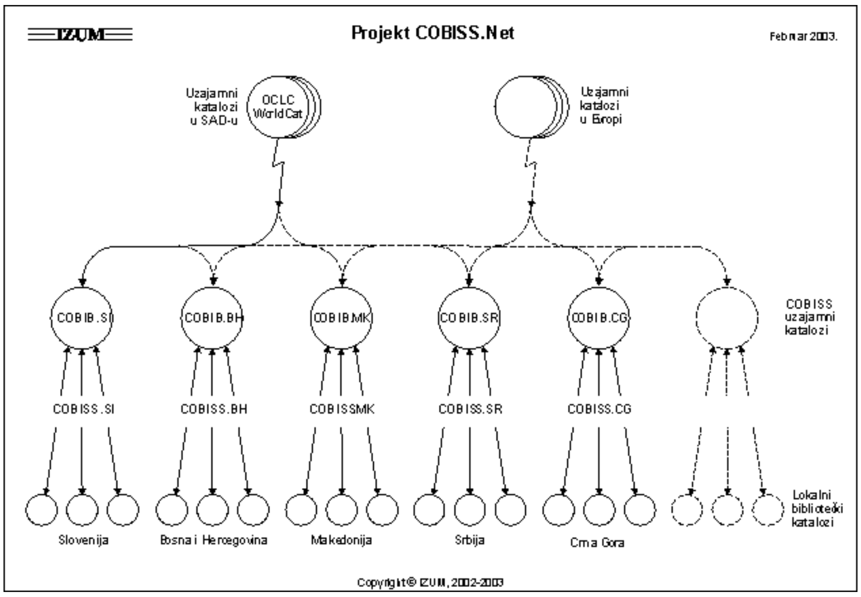
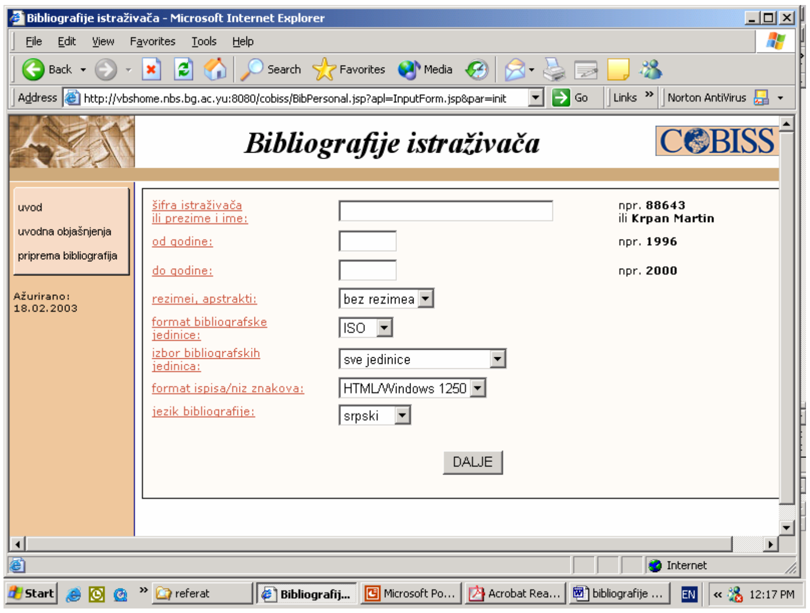


Figure 2. Schema of COBISS.Net project

Code list of scientific institutions enables the bibliographic output of these institutions that can exist as regular periodical publications instead of spending extra means and energy in producing them, as was the case until now.

Typology of document enables sorting of bibliographies according to type of scientific document. To manage bibliographies within the COBISS system, bibliographic items are sorted by valid typology of document/works. It is authors'/researchers' responsibility to accurately classify their items by typology. Displayed may only be bibliographic items from the COBIB database that contain either the researcher's code, or the evidence code given to the author. If certain items are missing yet retrievable from the COBISS/OPAC system by the researcher's last name and first name, the relevant library should be notified to add the author's/researcher's code to these items.

User chooses alone the desired parameters for bibliographic printout setting the time period intended to be covered by bibliography, abstracts and one of the displayed item formats and character sets. He can also decide whether some bibliographic items should be excluded from his personal bibliography.



Bibliografije istraživača

uvod
uvodna objašnjenja
priprema bibliografija

Ažurirano: 18.02.2003

šifra istraživača ili prezime i ime:		npr. 88643 ili Krpan Martin
od godine:		npr. 1996
do godine:		npr. 2000
rezimei, apstrakti:	bez rezimea	
format bibliografske jedinice:	ISO	
izbor bibliografskih jedinica:	sve jedinice	
format ispisa/niz znakova:	HTML/Windows 1250	
jezik bibliografije:	srpski	

DALJE

Figure 3. Bibliographic Item Format

One of the following display formats of bibliographic items can be chosen:

- ISI 690 –Documentation, Bibliographic references
- ISBD – based on IFLA standards,
- IEEE – form used in IEEE publications

HTML format is the default format for online display of bibliographies. One of the following sets of characters can be chosen: *Windows 1250*, *ISO 8559-2*, and *Unicode* (UTF-8). Standard bibliography display does not include abstracts. However, by using this parameter all abstracts or only abstracts in English may be displayed. An abstract will be displayed if it is part of the bibliographic record.

With connection of the shared catalogue to Journal Citation Reports Science Edition – JCR SE and Journal Citation Reports Social Sciences Edition – JCR SSE Impact factor of the journal where the scientific work has been published could be part of a personal bibliography.

A standard display contains all bibliographic items of a researcher within a selected period. Excluding of certain bibliographic items, Sorting of the list in ascending or descending order by the same parameters, and sending to mail addresses is included.

5. E-print University Repositories

The quality of a university is reflected by the quality of its intellectual output. Theses and dissertations reflect an institution’s ability to lead students and support original work. Electronic documents which are most frequently posted on university repositories are electronic theses and dissertations. Programme for establishment of the university repositories for electronic theses and dissertations has the healthy effect of helping a university to engage in issues related to scholarly communication.

Reasons and strategies for archiving electronic theses and dissertations can be summarized as following:

1) They make the results of graduate programmes widely known
2) Graduate programmes may be evaluated by the number of theses and dissertations produced and by the number of accessible electronic theses and dissertations
3) It is the easiest way to accomplish the goal of making theses and dissertations public
4) Theses and dissertations are referred by examining committees, so they have the necessary quality if to be published
5) They hold information that will help avoid duplication of efforts,
6) They require less storage space,
7) Programme for electronic theses and dissertations introduces digital libraries in the universities allowing other projects to develop
8) That is a way of sharing intellectual production
9) Such repositories give benefits to students, universities, regions/countries/ society
10) Free and easy access to information enhances the quality of theses and dissertations, since their authors and mentors know that they are easily accessible.

As the collection of available electronic theses and dissertations grows and reaches critical mass, it is likely that it will be frequently consulted by many researchers and graduate students from the universities in the country and abroad.

Building on the fact that students learn best by doing, introduction of the possibility to submit theses and dissertations in electronic format should encourage students to learn about electronic publishing and digital libraries. Students will produce higher quality work and faculty will demand better writing and clearer presentation of results Efforts to increase the information literacy are certain to benefit graduate students long after they have used these skills to produce a thesis or a dissertation.

With ETDs libraries can evolve into digital and online libraries. Because authors create and submit the digital documents and yet others validate them, libraries have the tasks to receive, store and provide access to them. Today many libraries in Serbia are already handling electronic journals. ETDs will give every library something unique in their digital resources.

6. Conclusion

The next step in building digital libraries as a part of higher education infrastructure should be collecting the data about main university professional and scientific production – theses and dissertations – and establishment of university repositories with open access.

Since the master theses and doctoral dissertations are even in Serbia nowadays produced mostly in electronic form it would not be impossible to start collecting them and build university repositories with standardized formats of theses and dissertations if necessary preparations are done and staff and equipment provided.

Software system of Digital Library of theses and dissertations was developed and deployed in test environment during the year 2003, at the University of Novi Sad. System is envisioned as an open archive and is supposed to be open for access to students, scientist and all other interested parties. The guidelines proposed by the NDLTD (Networked Digital Library of Theses and Dissertations) initiative are accepted and implemented.

Digital library of the Novi Sad University has been developed as an integral part of the BISIS system. It allows for metadata from the Digital library to be retrieved to the local databases. Other formats can also be used with the same protocol. The retrieval of complete bibliographic records could be achieved in that case. During the development of the Digital library system OAI Protocol for Metadata Harvesting has been implemented.

It is to be hoped that after initial testing all the records from the Digital library will be available to participants in the Virtual Library of Serbia system.

References

[1] Gordana Popović-Bošković, Stela Filipi-Matutinović : Virtuelna biblioteka Srbije (VBS) – računarsko povezivanje biblioteka u Srbiji. - INFOTEKA, god.2, br.1-2, str.57-80

[2] Miroslav Zarić : Implementation of protocol for metadata harvesting in networked digital library of theses and dissertations. - INFOTEKA, god.5, br.1-2, str. 99-112

[3] Dušan Surla, Zora Konjović, Branko Milosavljević...et.al.: Overview of implementation of the networked digital library of theses and dissertations. - INFOTEKA, god.5, br.1-2, str.75-86

[4] Mark Ware Consulting : PALS report on institutional Repositories. – Bristol, 2004

[5] Open Archives Initiative http://www.openarchives.org/

[6] Repository Explorer http://www.purl.org/NET/oai_explorer

[7] Academic Metadata Format http://amf.openlib.org/doc/ebisu.html

[8] Dublin Core http://dublincore.org/

Papers

Processing Bibliographic Documents in the Library Information System BISIS

Milan Vidaković[1], Tatjana Zubić[2], Branko Milosavljević[1],
Biljana Pupovac[3], Tatjana Tošić[2]
[1]Faculty of Technical Sciences, University of Novi Sad
[2]University of Novi Sad
[3]Faculty of Science and Mathematics, University of Novi Sad
{minja, tanja, mbranko, biljanap, ttosic}@uns.ns.ac.yu

Abstract. Bibliographical record editing is a central part of the library information system BISIS. Bibliographical record editing comprise the following operations on bibliographical records: bibliographic record search, creating new records, creating new records based on the existing ones, modification and deleting existing records. Record editing is based on the UNIMARC format of the record. The system also provides detailed reporting that icludes cataloguing cards, inventory book, bulletin of new publications, bibliography, etc. The system also supports distributed record search and download, which is provided by the shared cataloguing server.

1. Introduction

Bibliographical record editing [1, 2] is a central part of the library information system BISIS [3]. Editing of the bibliographical records is provided by a client application of the library information system. The client application provides the following operations on bibliographical records: bibliographic record search, creating new records, creating new records based on the existing ones, modification and deleting existing records. Record editing is based on the UNIMARC format of the record. An appropriate text server is used for work with records. In the BISIS system ver. 2 an Oracle ConText text server [5] was used and in the BISIS system ver. 3 a dedicated text server was developed. Records are saved in a relational database. The BISIS system ver. 3 text server does not depend on a specific relational system for database management – the system has been tested on several database servers: Oracle, SAP, Informix and MS SQL Server.

2. Bibliographical Editing Application

After the application is started, the librarian login window is invoked (figure 1). Librarian username and password are required. Both username and password are case sensitive. While password is being entered, each character is displayed as an asterisk (*).

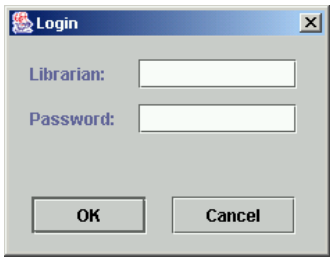


Figure 1. Librarian login window

Pressing the **OK** button starts the application setup. This setup depends on the librarian environment of the logged librarian. When the setup is complete, the search window is invoked (Figure 2).

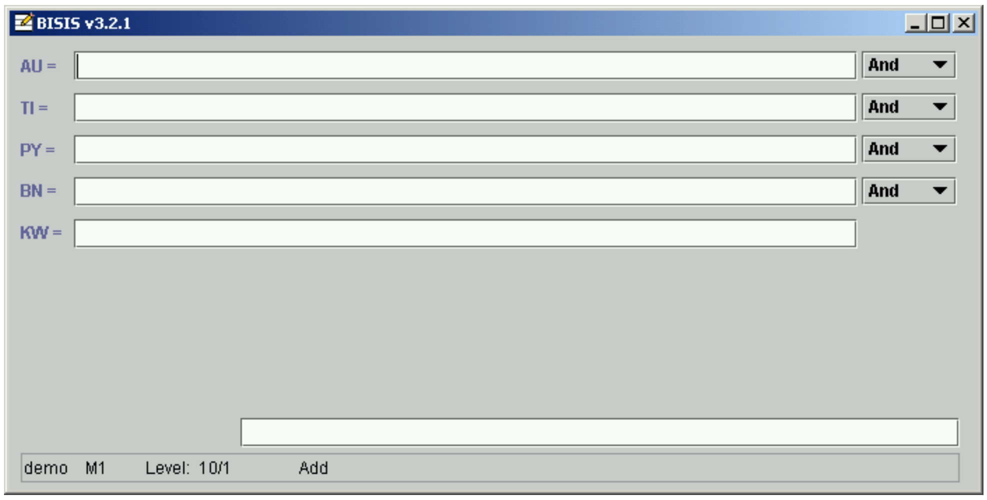


Figure 2. Application search window

Search window consists of the following elements:

- five search fields;
- command line field;
- status line field.

Each search field consists of the search prefix which can be chosen from the prefix code book, the search content text field and the operator drop-down list.

The ommand line displays number of hits (i.e. number of records which match search criteria) and enables textual command invocation.

The status line displays data about current editing status (librarian username, type of the publication, etc.).

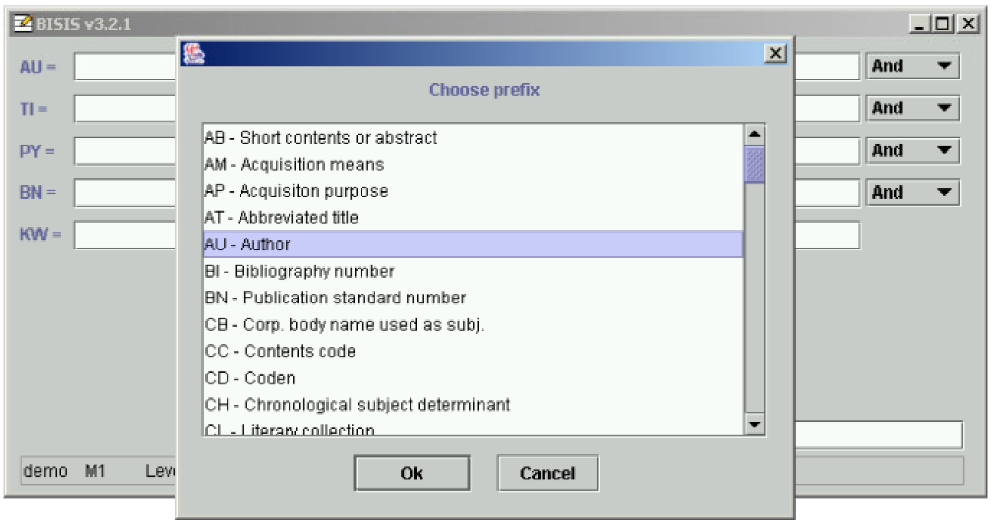
2.1. Search

Search can be done on the local or remote database. There are two types of search:

- basic search;
- search based on the **select** command.

2.1.1. Basic Search

There are five prefixes in the basic search. These prefixes depend on the librarian environment. On the right of the prefix name, there is a text field which is filled with the search text. If a librarian does not enter any text, no search will be started. The relationship between the prefixes is defined by the drop-down list that holds three operators: *And*, *Or* and *Not*. If the necessary prefix is not present among five predefined prefixes, it can be set from the prefix code book. The prefix code book is invoked by pressing the **<F9>** key while cursor is in the search content text field. The prefix code book window holds the list of the available prefixes (figure 3). There is a keyboard shortcut featuring the first letter of the prefix – if the key is typed, the programme will try to find a prefix starting with the pressed character.



AB - Short contents or abstract
AM - Acquisition means
AP - Acquisiton purpose
AT - Abbreviated title
AU - Author
BI - Bibliography number
BN - Publication standard number
CB - Corp. body name used as subj.
CC - Contents code
CD - Coden
CH - Chronological subject determinant
CL - Literary collection

Figure 3. Available prefixes in the prefix code book

After the prefix is selected, it is necessary to click the **OK** button (or press the **<Enter>** key). Alternatively, it is enough to do the double-click the prefix. Clicking the **Cancel** button or pressing the **<ESC>** key will cancel prefix setting.

The search text can contain wildcard characters ("*" and "?"). "*" character replaces zero or more characters, while "?" character replaces asingle character. To move the cursor between five search text fields, it is necessary to press the arrow keys (**<↑>** и **<↓>**) or to click the wanted field.

To find a record with the desired words or phrases, it is necessary to enter them in one or more search text fields (with optional use of wildcard characters – "*" and/or "?"). Phrases are entered as sequences of words. For example, if the following phrase is entered in the search text field next to the TI (title) prefix:

information systems journal

it will invoke the search which will look for the record having this sequence of words in the title. A query which has only the word **information** in the TI prefix field, will find all the documents having the word **information** in the title. There is no need to enter wildcards in a phrase, such as:

information *

The search is stared when the **<Enter>** key is pressed while the cursor is in either of the five search text fields. This will invoke the query interruption dialog which has the **Abort** button on it (Figure 4).

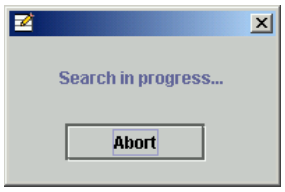
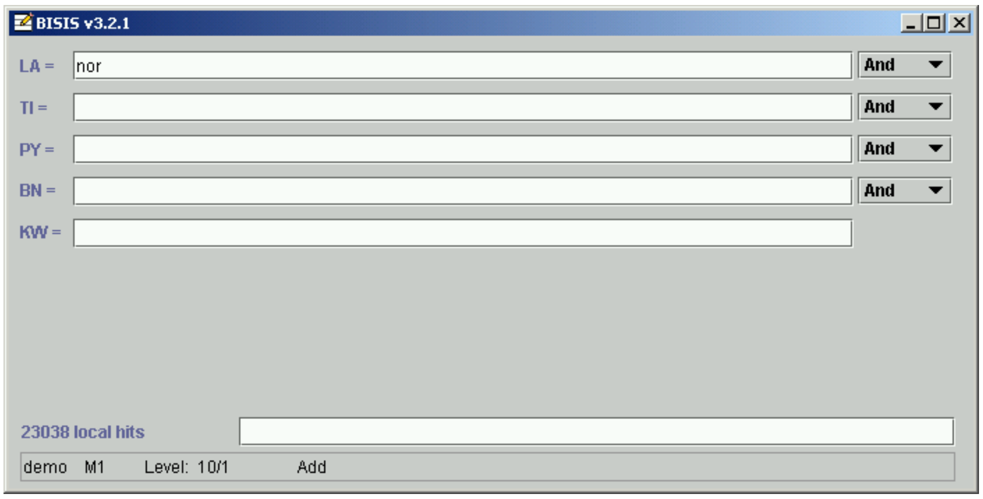


Figure 4. Query interruption dialog

When the search is over, the cursor will be placed in the command line which will have the number of hits in the database. If the **<ESC>** key is pressed, cursor is put back in the search text field.

Example 1. Find all records that have the Norwegian language of the text (Norwegian language is marked with the *nor* abbreviation). The *nor* abbreviation needs to be placed in the search text field right to the LA prefix. If the **<Enter>** key is pressed, the search is started. The search result (the number of records found) is displayed in the command line (Figure 5). The following command will display first three found records:

display 1-3



nor

23038 local hits

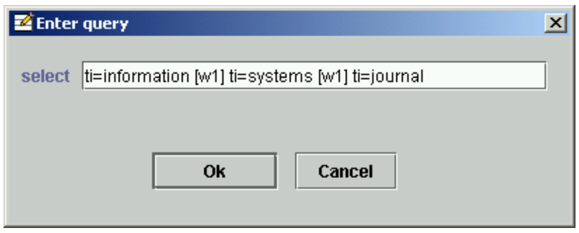
demo M1 Level: 10/1 Add

Figure 5. Search window after the search

2.1.2. Search Based on the SELECT Command

Search can be done using the **select** command of the *Dialog* language. The text of the **search** command is entered in a separate window (figure 6). This window is invoked when the **<F7>** key is pressed. To cancel entering **select** command, it is necessary to click the **Cancel** button or to press the **<ESC>** key. On the right of the **select** word there is a text field which will be filled with the query. A query entered in this field must be in compliance with the *Dialog* query language.

Example 2. Find all the records that have the *information systems journal* phrase in the title.



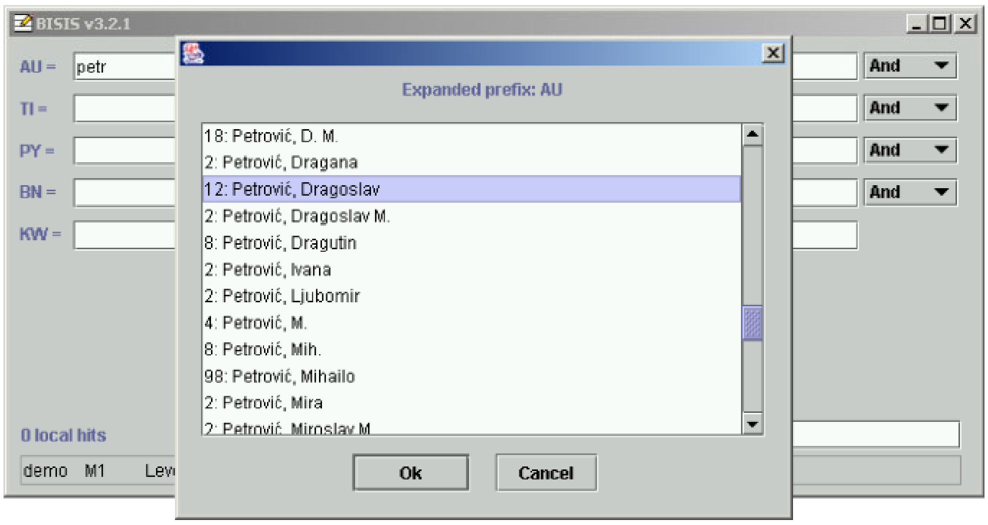
ti=information [w1] ti=systems [w1] ti=journal

Figure 6. **select** query window

When the search is over, the cursor will be in the command line, similar to the basic search.

2.1.3. Query Expansion (EXPAND)

If the <F8> key is pressed while the cursor is in one of the five search text fields, the list of all possible contents of the prefix that start with the entered text will appear. For example, if the word *petr* is entered in the AU search text field (no wildcards!) and **<F8>** is pressed, the window shown in the Figure 7 will appear.



```
18: Petrović, D. M.
2: Petrović, Dragana
12: Petrović, Dragoslav
2: Petrović, Dragoslav M.
8: Petrović, Dragutin
2: Petrović, Ivana
2: Petrović, Ljubomir
4: Petrović, M.
8: Petrović, Mih.
98: Petrović, Mihailo
2: Petrović, Mira
2: Petrović, Miroslav M.
```

Figure 7. List of all possible contents expanded from the entered text

This list holds the list of all existing terms (prefix contents) in the database which start with the entered text together with the number of occurrence. On the figure 4.7 there is a text *Petrović Miodrag* that exists in six records as an AU prefix content. If some of the items in the list are selected and **OK** button is pressed, the item text is copied into the search text field where **<F8>** key was pressed. This text can be used to search the database.

2.1.4. Remote Database Search

There is one condition for the remote database search – to have the shared cataloguing server running on the remote computer. When the shared cataloguing server is started, remote users can connect to it using the following command:

connect <internet_address>

The **internet_address** parameter is the internet address of the computer that has shared cataloguing server started.

If the connection to the server is successful, all the work is the same as with local database, with the following restriction: it is not possible to modify or delete remote records. To obtain the remote record, it is necessary to find it in the remote database and to transfer it from the remote database using the following command:

new <number_of_the_hit>

This command creates a new local record which is based on the remote record. The **number_of_the_hit** parameter is the number of the hit on the remote database. All further editing is done in the local database.

To stop working with the remote database, following command should be entered:

connect internal

This command disconnects the remote shared cataloguing server and connects to the local database.

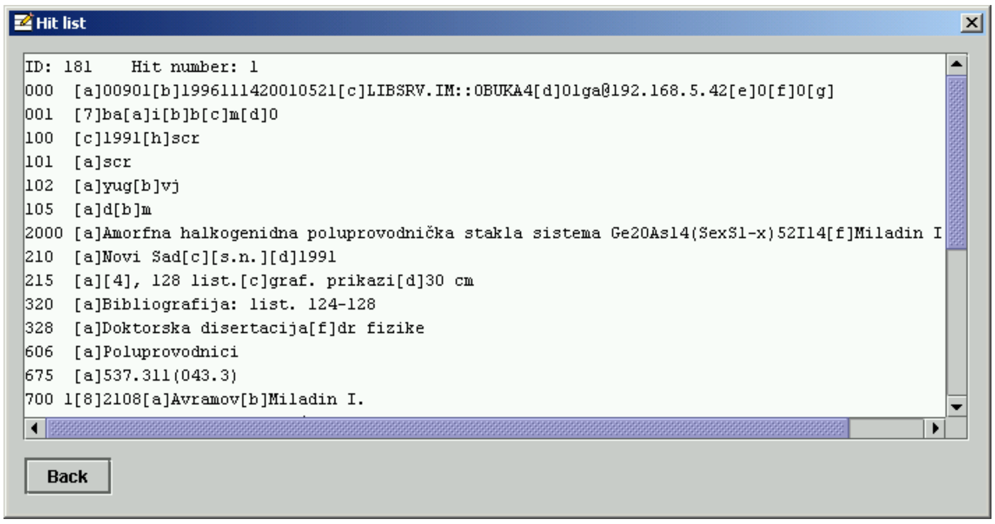
2.2. Record Maintenance

When the search is complete and certain records found, the following actions can be performed:

- display records found in the database;
- create a new record;
- create a new record based on an existing one (local or remote);
- modify the existing record;
- return to the search window.

Display records that are found in the database. To display records found in the database the user must enter following commands: **display** or **list**. The **display** command displays records in the current format, or one of the existing formats, or in a user-defined format. All existing formats can be listed using the following command: **format ?**. To change format or create a new one, a librarian must use the format command with appropriate parameters. The format command will be described later.

Example 3. After a successful search, **display 1** command will display the first hit in the current format (Figure 8). Figure 8 displays the FULL format as the current format. FULL format displays all UNIMARC elements of a record.



```
ID: 181    Hit number: 1
000  [a]00901[b]1996111420010521[c]LIBSRV.IM::OBUKA4[d]Olga@192.168.5.42[e]0[f]0[g]
001  [7]ba[a]i[b]b[c]m[d]0
100  [c]1991[h]scr
101  [a]scr
102  [a]yug[b]vj
105  [a]d[b]m
2000 [a]Amorfna halkogenidna poluprovodnička stakla sistema Ge20As14(SexS1-x)52I14[f]Miladin I
210  [a]Novi Sad[c][s.n.][d]1991
215  [a][4], 128 list.[c]graf. prikazi[d]30 cm
320  [a]Bibliografija: list. 124-128
328  [a]Doktorska disertacija[f]dr fizike
606  [a]Poluprovodnici
675  [a]537.311(043.3)
700 1[8]2108[a]Avramov[b]Miladin I.
```

Figure 8. Full format of the display command

Create a new record. To create a new record, the user must enter the **new** command. When a **new** command is entered, the editing parameters (publication type, processing level and mandatory level) are obtained from the librarian environment. After that, the editing window is invoked.

Example 4. When the **new** command is executed, the editing window is invoked (figure 9). Fields and subfields displayed in the window correspond to the current type of the publication (M1 – monographic publications), processing level (10 – Catalogue editing) and mandatory level (1 – Mandatory according to the UNIMARC standard). **Create a new record based on an existing one.** When the local or remote database is searched and at least one record is found, a new record based on the found one can be created. To achieve that, following command must be executed:

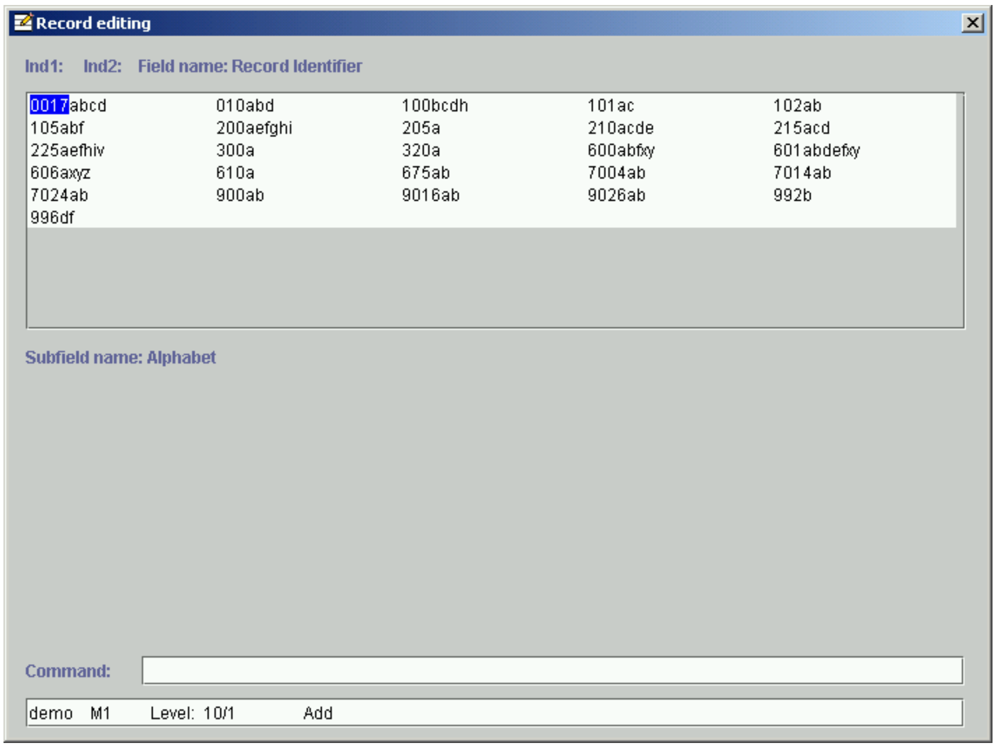
new <number_of_the_hit>

When the command above is executed, the editing window will appear. Content of this window is similar to the editing window invoked by the **new** command. All fields and subfields from the found record appear in this window except block 9 fields which have blank content. Also, additional fields and subfields are added to the new record, according to the librarian environment.

Modify of the existing record. An existing record can be modified using the following command:

edit <number_of_the_hit>

When the command above is invoked, the editing window will appear. Fields and subfields from the record appear in this window together with fields and subfields that correspond to the publication type, processing and mandatory levels which are defined in the librarian environment. The editing window is displayed in the Figure 10.


Record editing

Ind1: Ind2: Field name: Record Identifier

```
0017abcd        010abd          100bcdh         101ac           102ab
105abf          200aefghi       205a            210acde         215acd
225aefhiv       300a            320a            600abfxy        601abdefxy
606axyz         610a            675ab           7004ab          7014ab
7024ab          900ab           9016ab          9026ab          992b
996df
```

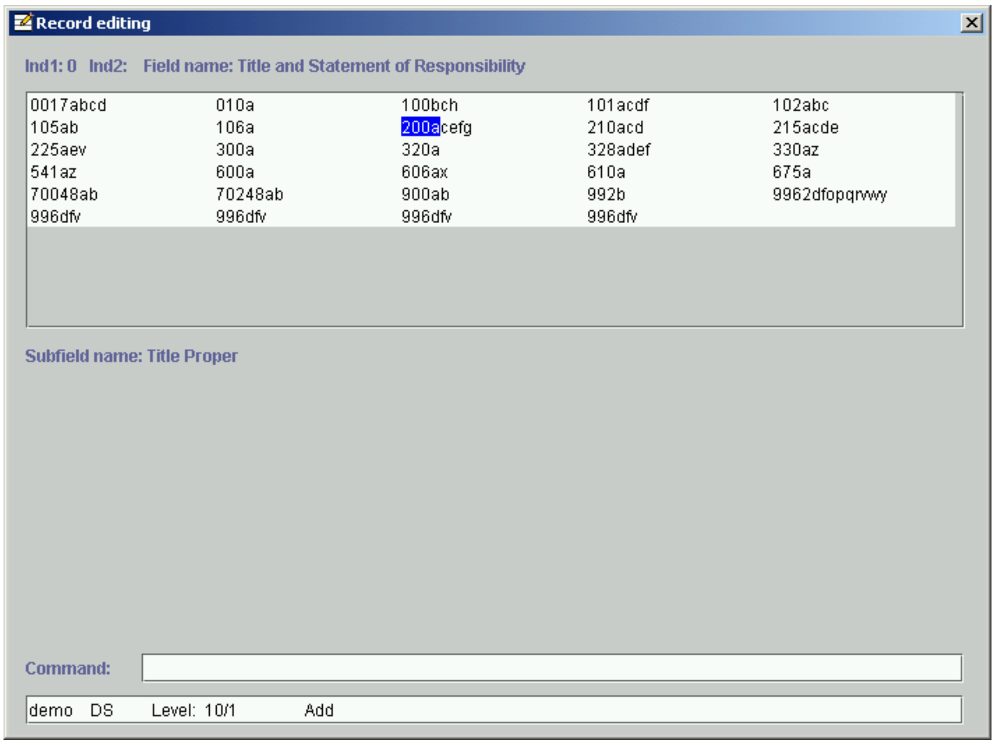
Subfield name: Alphabet

Command:

```
demo   M1      Level:  10/1           Add
```

Figure 9. Editing window after the **new** command is invoked

Return to the search window. To return to the search window the **<ESC>** key must be pressed. When the **<ESC>** key is pressed, the confirmation dialog will appear asking the user to save the current record. If the **Yes** button is clicked record will be saved and the programme will return to the search window. If the **No** button is selected, record will not be saved and the program will return to the search window. If the Cancel button is selected, the program will return to the editing window.



```
Ind1: 0   Ind2:   Field name: Title and Statement of Responsibility

0017abcd     010a       100bch     101acdf    102abc
105ab        106a       200acefg   210acd     215acde
225aev       300a       320a       328adef    330az
541az        600a       606ax      610a       675a
70048ab      70248ab    900ab      992b       9962dfopqrwy
996dfv       996dfv     996dfv     996dfv

Subfield name: Title Proper

Command:

demo   DS     Level:  10/1          Add
```

Figure 10. Editing window which is invoked by the **edit** command

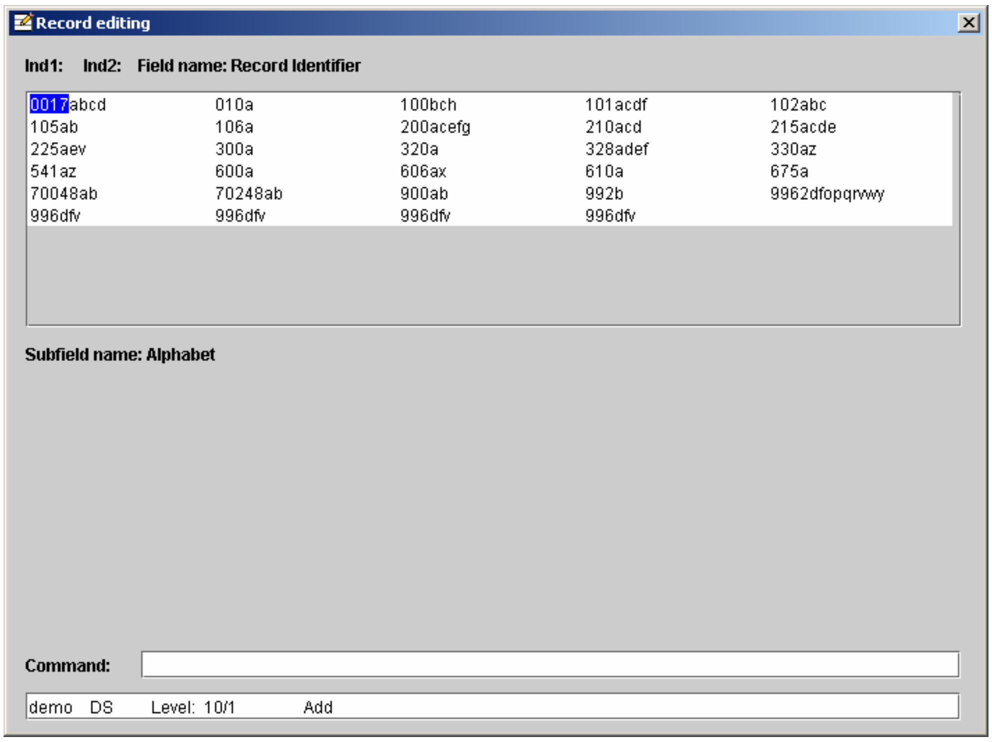
2.3. Record Editing

Modification of either existing or a new record is done in the editing window (Figure 11). Editing window consists of the following elements:

- first message line;
- directory;
- second message line;
- data editor;
- command line;
- status line.

A directory is the element of the window which is used to display record structure, i.e. fields and subfields of the record.

The first message line displays messages informing about the state of the first and second indicator and the name of the currently selected field in the directory.



Record editing

Ind1: Ind2: Field name: Record Identifier

```
0017abcd      010a          100bch        101acdf       102abc
105ab         106a          200acefg      210acd        215acde
225aev        300a          320a          328adef       330az
541az         600a          606ax         610a          675a
70048ab       70248ab       900ab         992b          9962dfopqrvwy
996dfv        996dfv        996dfv        996dfv
```

Subfield name: Alphabet

Command:

```
demo   DS      Level: 10/1          Add
```

Figure 11. Editing window

The second message line displays the name of the current subfield in the directory.

The data editor is invoked when the appropriate field and subfield are selected and the <Enter> key is pressed. There are two types of the data editor: single-line and multi-line.

The command line is used for entering commands.

The status line displays the same data as the status line of the search window.

2.4. The Directory

The directory is a part of the editing window which displays the record structure, i.e. record fields and subfields. To move between fields, it is necessary to press the cursor keys (<↑>, <↓>, <→> and <←>**).** The current field and subfield are highlighted (see Figures 9, 10, and 11). The directory offers the following actions:

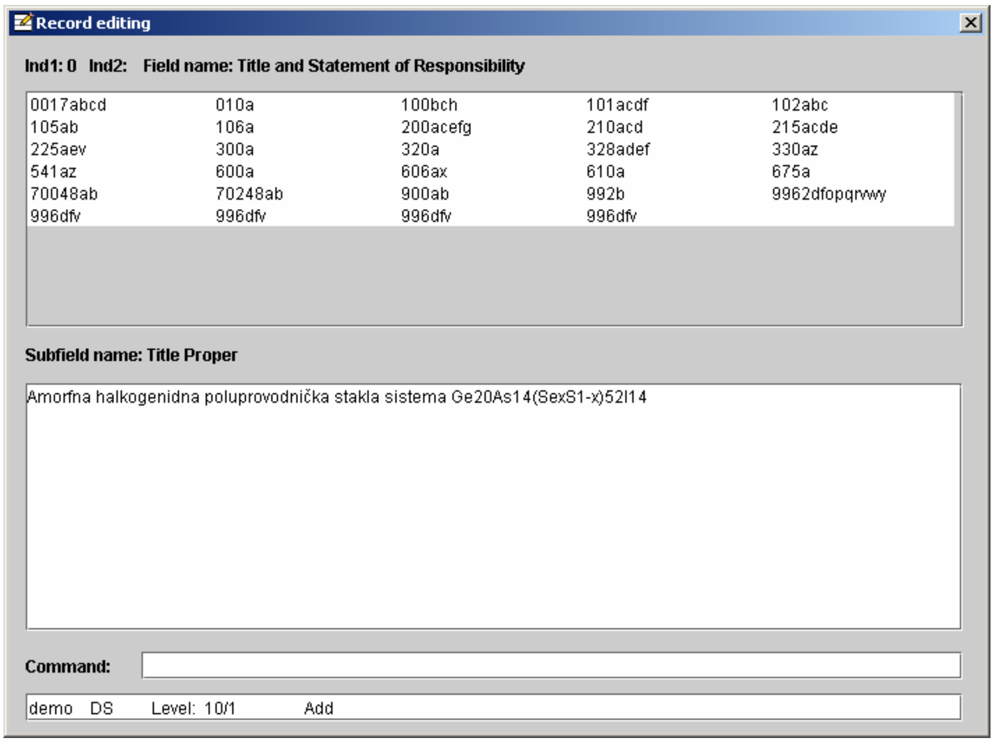
- subfield content entering;
- assigning indicator value;
- adding a field;
- deleting a field;
- undeleting a field;
- adding a subfield;
- deleting a subfield;
- undeleting a subfield;
- entering subsubfields;
- entering secondary fields;
- saving the record;
- quitting without saving the record;
- changing the publication type, processing and mandatory levels;
- command entering.

Subfield content entering. In order to enter the subfield content, it is necessary to position on the appropriate field and subfield. To position on the field user must use cursor keys. To position on the subfield, user must use **<KP1>** and **<KP3>** keys. If the <Enter> key is pressed, the data editor is invoked. The data editor can be a single-line (e.g. to enter a piece of the coded information), or multi-line (e.g. the content of the subfield is a text of arbitrary length). The content is stored in the subfield if the **<F12>** key is pressed (Figure 12). If the <**ESC**> key is pressed storing is cancelled.

If the subfield content is coded, from the single-line data editor it is possible to invoke the appropriate code book by pressing the <F9> key. The value from the code book can be transferred to the data editor by selecting the appropriate code and clicking the OK button (figure 13), or by pressing the <Enter> key. Also, a doubleclick on the code will transfer it to the data editor. If the Cancel button is clicked, or <ESC> key is pressed, the code will not be transferred.

Assigning indicator value. The indicator value can be chosen or changed the following way: when the appropriate field is highlighted in the directory, pressing the **<KP2>** key will invoke the indicator window. Indicator window can assign the first and second indicators to the field if they exist (figure 14). If either of the indicators does not exist, the drop-down list displaying its value will be inaccessible. If both indicators do not exist, there the appropriate message will be displayed.

By clicking the **OK** button, selected indicator values are transferred to the highlighted field. If the **<ESC>** key is pressed or **Cancel** button clicked, indicators are not changed.



Ind1: 0 Ind2: Field name: Title and Statement of Responsibility

0017abcd	010a	100bch	101acdf	102abc
105ab	106a	200acefg	210acd	215acde
225aev	300a	320a	328adef	330az
541az	600a	606ax	610a	675a
70048ab	70248ab	900ab	992b	9962dfopqrwy
996dfv	996dfv	996dfv	996dfv	

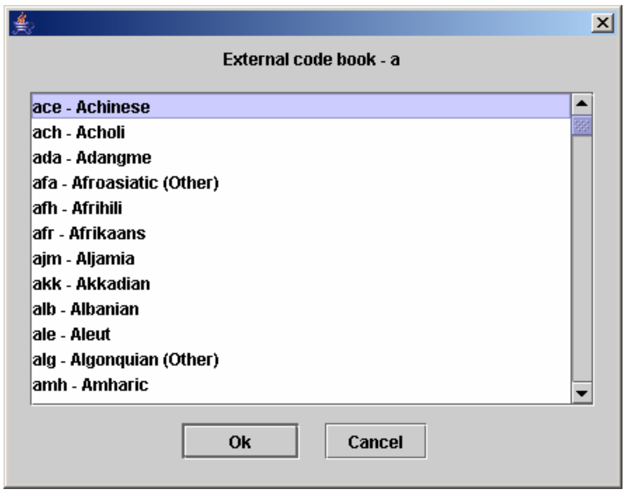
Subfield name: Title Proper

Amorfna halkogenidna poluprovodnička stakla sistema Ge20As14(SexS1-x)52I14

Command:

demo DS Level: 10/1 Add

Figure 12. Entering the subfield content in the multi-line data editor


External code book - a

ace - Achinese
ach - Acholi
ada - Adangme
afa - Afroasiatic (Other)
afh - Afrihili
afr - Afrikaans
ajm - Aljamia
akk - Akkadian
alb - Albanian
ale - Aleut
alg - Algonquian (Other)
amh - Amharic

Figure 13. Choosing the appropriate code from the code book

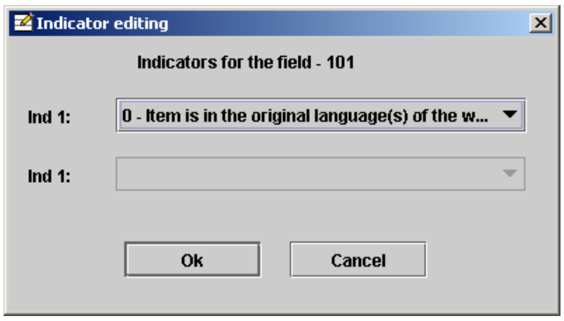

Indicator editing
Indicators for the field - 101
Ind 1:
0 - Item is in the original language(s) of the w...
Ind 1:
Ok
Cancel

Figure 14. Choosing indicator values

Adding a field. If the <KP7> key is pressed, Add field dialog is invoked (figure 15). The upper part of the window displays selected fields to be added in the record (in Figure 15, it is field 600). The bottom part of the window has two lists: first has all available fields to be added at the current position (depending also on the **Insert**/**Add** mode), and the second has the list of all available subfields for the selected field (it is possible to select more than one subfield). When the appropriate field and subfield(s) are selected they are transferred to the upper part of the window using the **Add** button. Button **Remove** removes field and subfield(s) from the upper part of the window. The field and subfield(s) are put into the directory using the **OK** button.

If the mode is **Insert**, then the field and subfield(s) are inserted before the highlighted field. If the mode is **Add**, the field and subfield(s) are added after the highlighted field. Switching between **Insert**/**Add** mode is done using the **<Insert>** key. To cancel adding a field user needs to click the **Cancel** button or to press **<ESC>** key.

Deleting a field. A field can be deleted if it is highlighted and the **<KP9>** key is pressed.

Undeleting a field. A previously deleted field is undeleted by pressing **<Ctrl>+<KP9>** key combination. Depending on the mode, the deleted field will be returned before the highlighted field (**Insert** mode) or after (**Add** mode). If the deleted filed is repetitive, it can be multiplied with subsequent pressing the **<Ctrl>+<KP9>** key combination.

Adding a subfield. A subfield can be added by pressing the **<KP4>** key. It will invoke the window that will list all the available subfields that can be added to the current field (figure 16). If the mode is **Insert**, the new subfield will be inserted before the current subfield and if the mode is **Add**, it will be added after the current subfield. To **Add**/**Insert** a required subfield, it is necessary to select it and click the **OK** button or press the **<Enter>** key. Alternatively, it can be achieved by double-clicking the required subfield. To cancel adding and return to the directory, the user must click the **Cancel** button or press the **<ESC>** key.

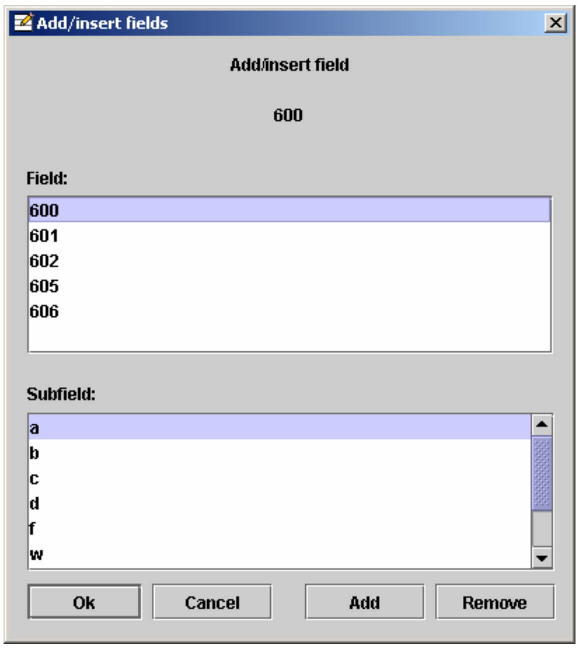


Figure 15. Adding a field

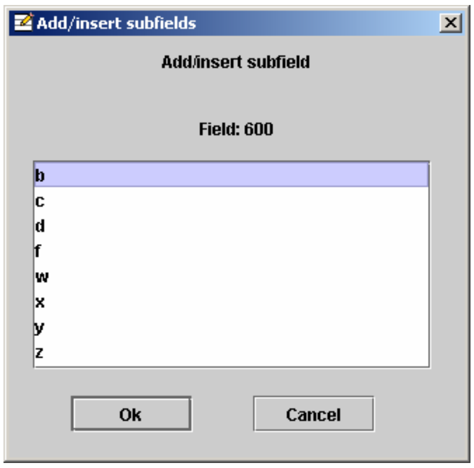


Figure 16. Adding a subfield

Deleting a subfield. A subfield can be deleted if it is selected and the <KP5> key is pressed.

Undeleting a subfield. A subfield which was deleted can be undeleted by pressing the **<Ctrl>+<KP5>** key combination. Depending on the mode, the deleted subfield will be inserted before the current subfield (**Insert** mode) or added after the current subfield (**Add** mode). If the deleted subfield is repetitive, it can be multiplied by subsequently pressing the **<Ctrl>+<KP5>** key combination.

Entering subsubfields. If a subfield contains subsubfields, then when the <Enter> key is pressed in the directory, the subfield dialog will appear. The subsubfields marked with an asterisk (the (*) character) already have the content entered. By selecting the appropriate subsubfield and clicking the Set button (or pressing the <Enter> key), the data editor will appear (single-line or mutli-line). The subsubfield content will be stored into the subsubfield by pressing the <F12> key or cancelled by pressing the <ESC> key. Subsubfield dialog is closed when the Exit button is clicked (or <ESC> key is pressed).

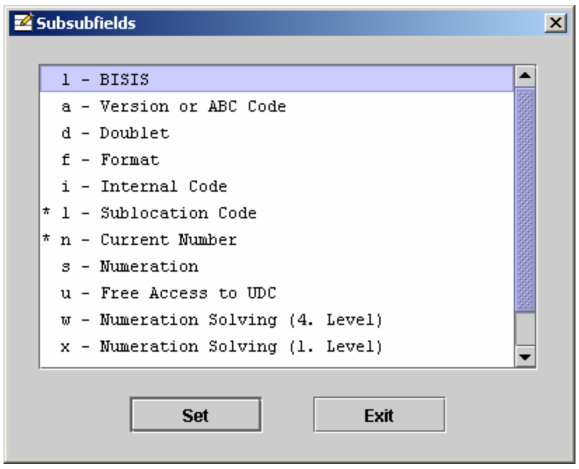
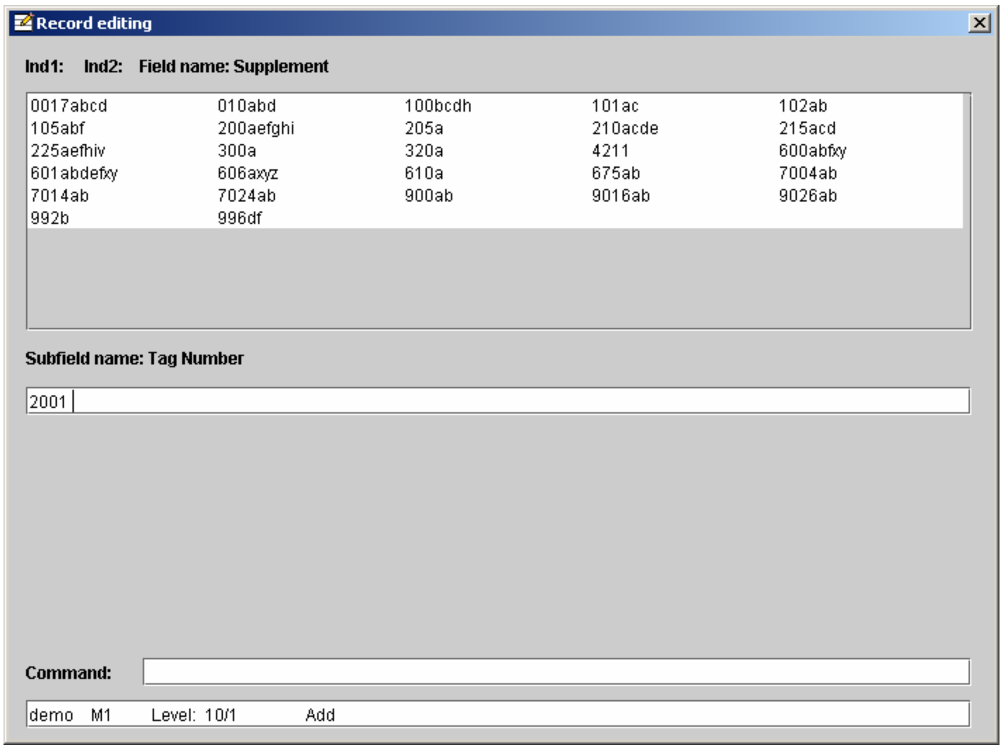


Figure 17. Adding a subsubfield

Entering secondary fields. Secondary fields are stored in the subfield 1 of the field 421, 421 or 469. When entering the subfield 1 content, the secondary field name and two indicators are entered. Figure 18 displays entering the secondary field 200 with the first indicator set to 1 (as a content of the subfield 1 in the field 421).

When the secondary field is stored in the subfield 1, it is possible to add secondary subfields only (figure 19). Secondary subfields are subfields that belong to the secondary field. Figure 20 displays the field 421 after adding secondary subfields. To enter a new subfield 1 it is necessary to set the data entering mode to **Insert** and to select the existing subfield 1. The newly created subfield will be inserted before the existing subfield. If the data entering mode is **Add**, the subfield 1 can be added before the existing subfield 1 is filled with data.


Record editing

Ind1: Ind2: Field name: Supplement

0017abcd	010abd	100bcdh	101ac	102ab
105abf	200aefghi	205a	210acde	215acd
225aefhiv	300a	320a	4211	600abfxy
601abdefxy	606axyz	610a	675ab	7004ab
7014ab	7024ab	900ab	9016ab	9026ab
992b	996df			

Subfield name: Tag Number

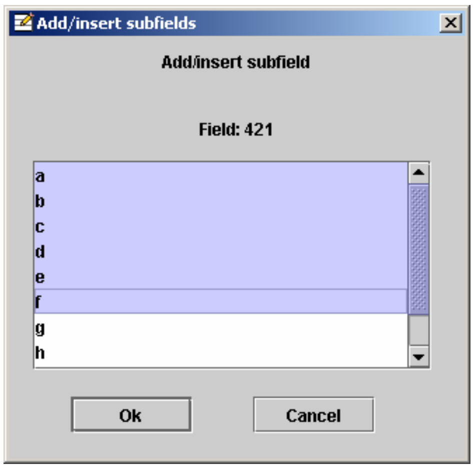
2001

Command:

demo M1 Level: 10/1 Add

Figure 18. Entering a secondary field

Saving the record. To save an edited record it is necessary to go to the command line (by pressing **<F6>** key). There are two commands which would save the record: **save** and **exit**. If the **save** command is entered, it will check whether all the required fields and subfields exist (depending on the mandatory level). Then, it will display a short overview of the record to be saved. If the **OK** button is pressed, the record will be saved. If the **Cancel** button is pressed, saving will be interrupted. If there are some missing fields and subfields, they will be displayed in the missing fields/subfields list. After the record is saved, the cursor remains in the command line. To return to the directory, the **<ESC>** key needs to be pressed. If the **exit** command is entered, it will perform the same action as the **save** command, and it will return from the editor window and position the cursor in the command line of the search window.

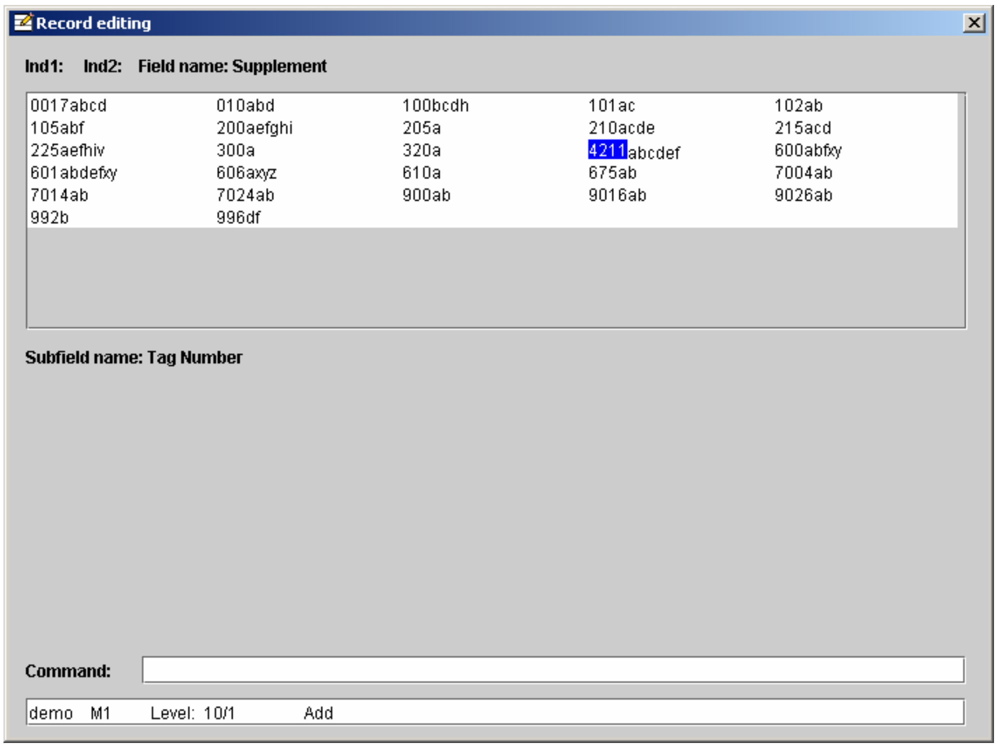


Add/insert subfield

Field: 421

a
b
c
d
e
f
g
h

Figure 19. Adding subfields to the secondary field



Ind1: Ind2: Field name: Supplement

```
0017abcd      010abd       100bcdh     101ac        102ab
105abf        200aefghi    205a        210acde      215acd
225aefhiv     300a         320a        4211abcdef   600abfxy
601abdefxy    606axyz      610a        675ab        7004ab
7014ab        7024ab       900ab       9016ab       9026ab
992b          996df
```

Subfield name: Tag Number

Command:

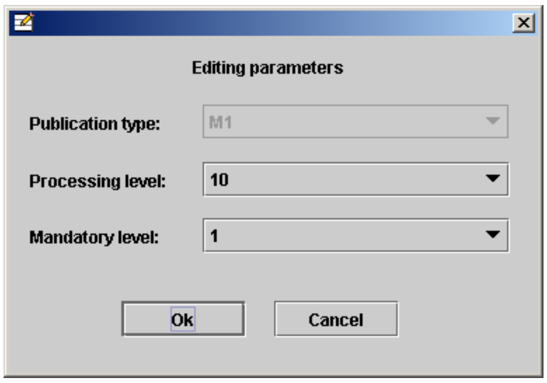
demo M1 Level: 10/1 Add

Figure 20. Secondary field with subfields

Saving the record is a background activity. During record saving, it is possible to resume work. While saving is in progress, the appropriate message is displayed in the status line: **Saving in progress**. As long as this message remains in the status line, user should not quit the application. Otherwise, the record being saved may be damaged.

Quitting without saving. If the librarian does not wish to save the record, there are two ways of avoiding it: pressing the <ESC> key in the directory, or issuing the quit command from the command line. In both cases, there will be a confirmation dialog asking the user to confirm quitting. If the OK button is pressed, application will return the search dialog without saving.

Changing the publication type, processing and mandatory levels. The status line displays the current publication type, processing and mandatory levels for the logged-in librarian. If the librarian is in the search window, it is possible to change all three parameters by pressing the **<F12>** key. If the user is editing the record, it is possible to change processing and mandatory levels only. It can be done by pressing the same **<F12>** key, and it will invoke the appropriate window (Figure 21).


Editing parameters

Publication type: M1

Processing level: 10

Mandatory level: 1

Figure 21. Changing the processing and mandatory level

Entering commands. To switch to the command line it is necessary to press <F6> key. To return to the directory <ESC> key needs to be pressed.

3. Commands

There are two groups of commands:

- Commands for record manipulation,
- Reporting commands.

Following section gives the list of the all available commands including their formats, abbreviations (if they exist) and descriptions.

3.1. Commands for Record Manipulation

This group of commands includes commands for a record creation, modification and deleting.

Command: **delete**
Abbreviation: **del**
Format: **delete**

Description: This command deletes the record in the database. The librarian will be warned that the record will be deleted. This action can be accepted (the **Yes** button) or cancelled (the **No** button).

Command: **edit**
Abbreviation: **e**
Format: **edit** <number_of_the_hit>

Description: This command opens the record processing window for the chosen record.

Command: **new**
Abbreviation: **n**
Format: **new** [number_of_the_hit]

Description: If the number of the hit is not given, a new record will be created according to the current publication type, the processing and mandatory levels. If the number of the hit is given, a new record will be created based on the chosen hit, i.e. the content of the chosen hit will be copied into the new record.

Command: **save**
Abbreviation: **s**
Format : **save**

Description: The **save** command performs saving the record.

Command: **exit**
Abbreviation: **ex**, **x**
Format : **exit**

Description: The **exit** command performs exiting the record with saving it into the database.

Command: **quit**
Abbreviation: **q**
Format : **quit**

Description: The **quit** command performs exiting the record without saving it into the database.

3.2. Commands Used for Reporting and Documenting

When reporting and documenting, the following commands are used:

Command: **display**
Abbreviation: **d**
Format: **display** <range>

Description: This command is used for displaying records in the currently chosen format. The records to be displayed can range between one hit and the interval of hits. The interval of hits is given by stating the first and the last hits separated by a hyphen. For example,

display 4

will display the hit number 4, while

display 6-10

will display the hits from the ordinal number 6 to the ordinal number 10.

Command: **format**
Abbreviation: **f**
Format: **format** < the_format_name >
format < the_format_name > < the_format_definition >
format full
format ?

Description: **format** command defines or chooses display format. If the parameter is the name of the format, it will be set as the current format. If there is another parameter, it will be considered as a format definition and the current format will be set to it. The format definition consists of the list of prefixes separated by the comma character. For example:

format moj_format AU,TI,KW

will define a new format with the name **moj_format** and it will set it to be the current format. The new format will display the three prefixes for each record (AU, TI and KW). The command:

format full

will set the display in the UNIMARC format to be the current one. The command:

format my_format

will set the format named **my_format** (an already defined format) to be the current one.

Command: **list**
Abbreviation: **l**
Format: **list** [<name_of_cataloguing_card>][,<range>]
list ?

Description: This command creates cataloguing card(s) for the given range. The records to be displayed can range between one hit and the interval of hits. The interval of hits is given by stating the first and the last hits separated by a hyphen. The command **list ?** lists all available types of the cataloguing cards. For example, the command:

list L41,1-10

will list cataloguing cards, type L41 for the first ten hits. Command:

list 1

will create cataloguing card of the default type for the first hit.

Command: **print**
Abbreviation:
Format: **print**

Description: Command **print** invokes reporting window that is used for report printing (Inventory book, Bulletin of new publications, Bibliographies, etc.).

4. Reporting

Reports are issued by entering the **list** command from the search window command line (Figure 22).

The **list** command opens the reporting window. This command can be invoked only after the search has been completed or while editing the record. Depending on the origin, this command has different parameters. If it is invoked after the search has been completed, it is necessary to enter hit number(s). If it is invoked while editing the record, no hit number is necessary. The second parameter (report type) is optional. If it is omitted, the report is created using the default report type obtained from the librarian environment which was set during the initialisation or manually set by the librarian.

Figure 22 displays the **list** command invocation from the search window. Figure 23 displays the result of the **list** command execution, which is a cataloguing card for the monographs.

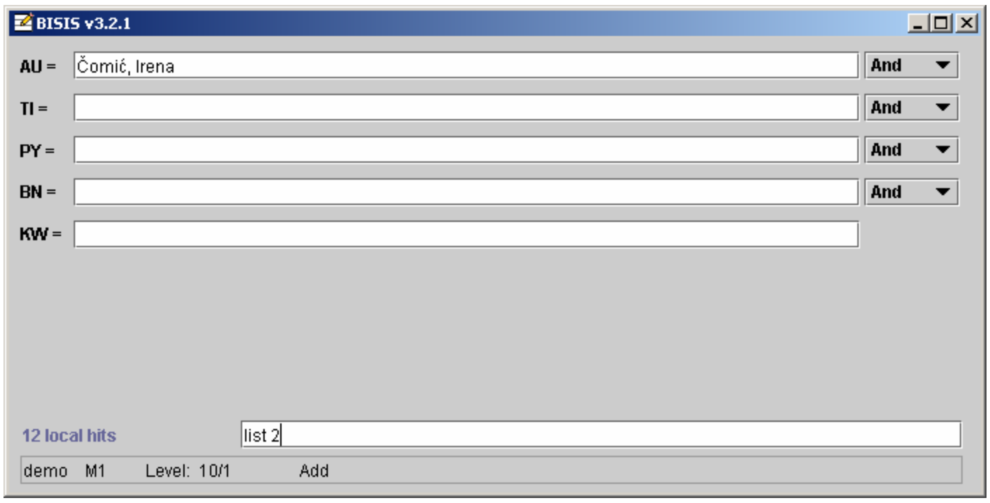
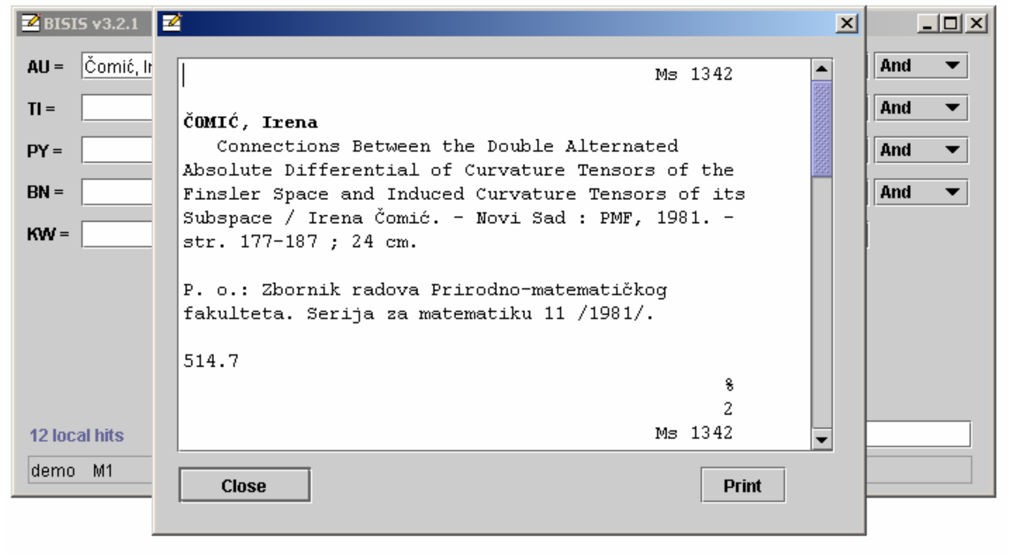


Figure 22. **list** command invocation

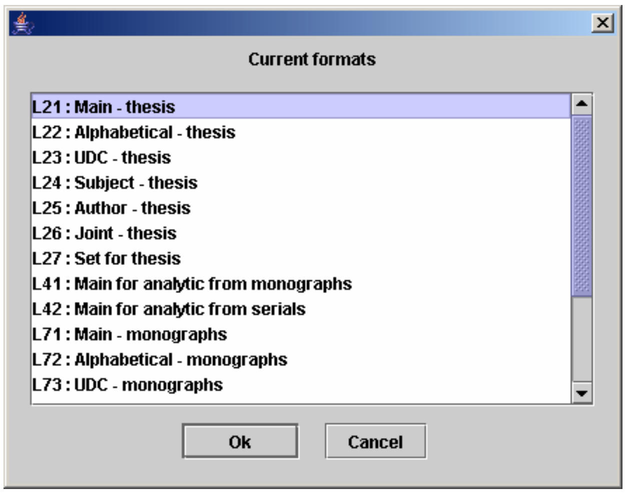


```
                                                    Ms 1342

ČOMIĆ, Irena
   Connections Between the Double Alternated
Absolute Differential of Curvature Tensors of the
Finsler Space and Induced Curvature Tensors of its
Subspace / Irena Čomić. - Novi Sad : PMF, 1981. -
str. 177-187 ; 24 cm.

P. o.: Zbornik radova Prirodno-matematičkog
fakulteta. Serija za matematiku 11 /1981/.

514.7
                                                         %
                                                         2
                                                    Ms 1342
```

Figure 23. Result of the **list** command execution

To print the report it is necessary to click the Print button.

The **list ?** command displays the list of all available report types. Figure 24 displays the window containing all available report types.


Current formats

L21 : Main - thesis
L22 : Alphabetical - thesis
L23 : UDC - thesis
L24 : Subject - thesis
L25 : Author - thesis
L26 : Joint - thesis
L27 : Set for thesis
L41 : Main for analytic from monographs
L42 : Main for analytic from serials
L71 : Main - monographs
L72 : Alphabetical - monographs
L73 : UDC - monographs

Figure 24. Result of the **list ?** command execution

The BISIS system has the following cataloguing card types implemented: alphabetical, joint, and secondary (subject, author and UDC) cataloguing cards for monographs, serials, thesis, cartographic and non-printed materials.

Also, the following reports are implemented: inventory book, bibliography, bulletin of new publications. The **print** command creates reports mentioned above. This command invokes the print window (displayed in the Figure 25).

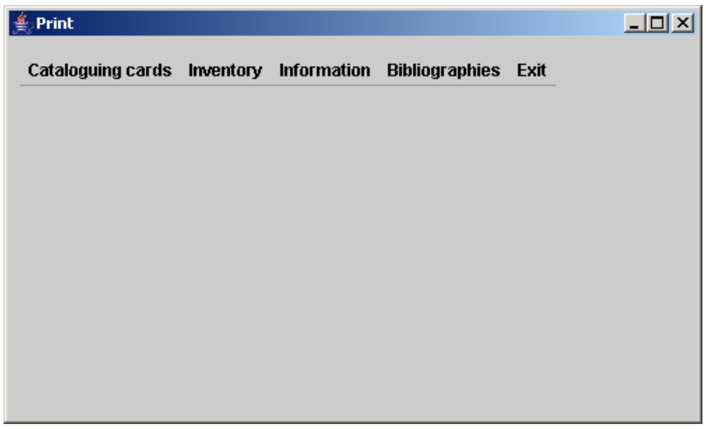
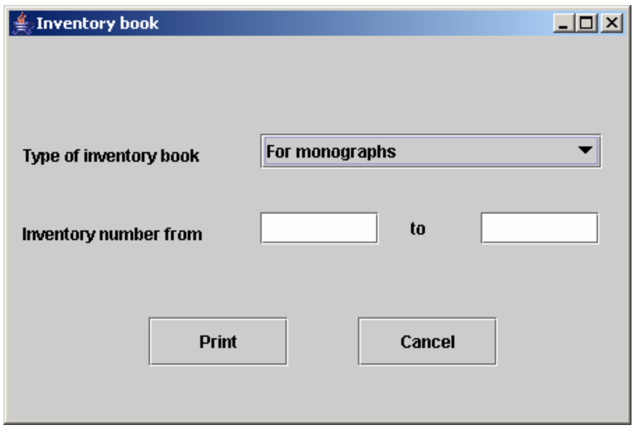


Figure 25. Reports menu

Inventory book is printed by choosing the **Inventory book** option from the **Inventory** menu. This option invokes the inventory book window. The user can choose the type of the inventory book and choose the range of records to be included in the inventory book (Figure 26). The **print** button invokes the preview window which prints the inventory book.

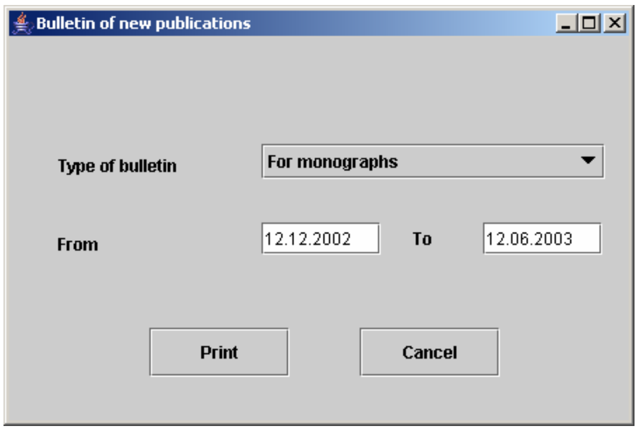


Type of inventory book: For monographs

Inventory number from: to

Figure 26. Inventory book dialog

Bulletin of new publications is invoked by choosing the Bulletin of new publications option from the Information menu. This option opens the window (Figure 27) consisting of the type of the bulletin and the date range.



Type of bulletin: For monographs

From: 12.12.2002 To: 12.06.2003

Figure 27. Bulletin of new publications

Bibliography depends on records found in the records search. Only records found in the database can be used to print the bibliography. When the number of hits is greater that zero, Bibliography option from the Bibliographies menu opens the bibliography window (Figure 28) which consists of the sorting criteria (by author, title and publication year, for example). The **Print** button opens the preview window which prints the bibliography.

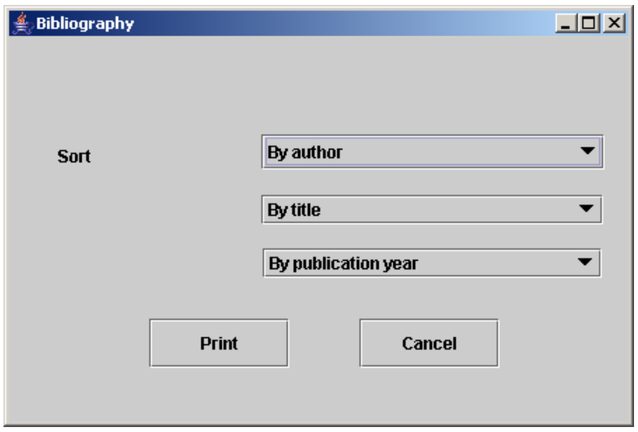


Figure 28. Type of bibliography sort window

The Unicode standard enables report printing in different languages and alphabets including Cyrillic, Latin and Greek letters.

5. Directory Keyboard Commands

The following keyboard commands are available in the directory:

- keys <←>, <→>, <↑>, <↓> move selection through fields;
- keys <**KP1**> (or <**Shift+**←>) and <**KP3**> (or <**Shift+**→>) move selection through subfields;
- <**Enter**> – invokes data editor;
- <**KP2**> – invokes indicator dialog;
- <**Insert**> – changes Add/Insert mode;
- <**ESC**> - quits editing;
- <**KP7**> - adds field;
- <**KP9**> - deletes field;
- <**Ctrl**>+<**KP9**> - undeletes field;
- <**KP4**> - adds subfield;
- <**KP5**> - deletes subfield;
- <**Ctrl**>+<**KP5**> - undeletes subfield;
- <**F6**> - moves to the command line;
- <**F12**> - changes processing and mandatory level for the current type of the publication;
- <**F9**> - lists all available fields and subfields for the current type of the publication;

Note: <**KP0**> to <**KP9**> are numpad keys from 0 to 9.

6. Conclusion

The client application of the BISIS library information system provides simple and efficient library record maintenance. Library records are based on the UNIMARC standard and are stored in a RDBMS. Application does not depend on a particular RDBMS. The application is multilingual on several levels:

- the application is internationalised and localised for several languages and
- the record modification based on the UNIMARC standard provides record search independent from the original alphabet.

The application also supports distributed record search and download, which is provided by the shared cataloguing server.

References

[1] Vidaković, M., Implementacija korisničkog interfejsa na bibliotečkom informacionom sistemu u Java okruženju, Book of abstracts SIFON'97, Zlatibor 1997., pp. 10.

[2] Vidaković, M., *Object oriented specification and implementation of library information system user interface*, Master thesis, Faculty of engineering, Novi Sad 1998., http://diglib.ns.ac.yu/ndltd/docs/set1/ndltd7/MinjaMagistarski.PDF

[3] Surla, D., Konjović, Z., Pupovac, B., Milosavljević, B., Vidaković, M., Tošić, T., Zubić, T., *Uputstvo za korišćenje bibliotečkog softverskog sistema BISIS ver. 3*, Grupa za Informacione tehnologije, Novi Sad 2004.

[4] Todorić, D., Vulić, T., *Data structure of UNIMARC standard*, Proceedings of the X Conference on Applied Mathematics PRIM'95, Budva 1995, pp. 309-314.

[5] Milosavljević, B., Konjović, Z., *Tekst server bibliotečkog informacionog sistema zasnovan na Oracle ConText Option-u*, Book of abstracts SINFON'97, Zlatibor 1997, pp. 7.

[6] Milosavljević, B., *Text server for UNIMARC records*, Master thesis, Faculty of engineering, Novi Sad 1999., http://diglib.ns.ac.yu/ndltd/docs/set1/ndltd6/textsrv.pdf

Maintenance of the YUMARC Format

Biljana Pupovac
Faculty of Science and Mathematics, University of Novi Sad
biljanap@uns.ns.ac.yu

Abstract. Library information system BISIS ver. 3.0 supports library operation and is based on the YUMARC format. This paper gives a detailed description of the use of the application for modifying YUMARC format, which is an integral part of the BISIS software system.

1. Introduction

The MARC bibliographical format (*Machine Readable Cataloguing*) [1], as one of the first formats for exchanging bibliographical data and other related data in the machine readable form, has served as the basis for defining other bibliographical MARC formats which have been intended to meet the needs of different countries and library systems. Thus, UNIMARC (*Universal Machine Readable Cataloguing*) [2] has been developed to satisfy the needs of the European countries. The UNIMARC format served as the foundation stone for some national bibliographical formats. The YUMARC format completely reflects the basic structure and the data scope of the UNIMARC format, while at the same time being enhanced for the national use and adapted to the BISIS software system. Ther paper [3] presents the overview of bibliographical formats development.

UNIMARC is a universal bibliographical material cataloguing format for recording bibliographical data in the machine readable form and it enables data exchange between national and international bibliographical institutions. This format defines the structure of the bibliographical description of library units. A bibliographical description of a library unit formed according to the UNIMARC format is called *a UNIMARC record* or *a biliographical record*. A set of bibliographical records makes *bibliographical material*. According to the UNIMARC format, a record contains record indicators and a number of fields (grouped in blocks). A field contains a set of subfields, and it also has two associated indicators. Subfields content is data about the publication being catalogued. The meaning of the datum is unambiguosly defined by the field, associated indicators and the containing subfield.

The library software system BISIS ver. 3.0 [4] has been developed at the University of Novi Sad. It is based on the YUMARC format. This paper presents a detailed description of the use of the application for YUMARC format modification.

2. Application for YUMARC Format Modification

The application for YUMARC format modification [5, 6] has been implemented within the BISIS software system version 3.0. It has been done in the Java environment [7].

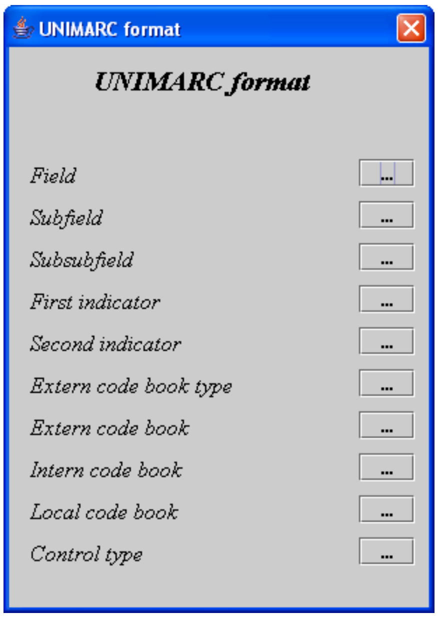


Figure 1. Starting window

The starting window for YUMARC format modification is shown in Figure 1. This window enables moving on to the concrete window for modifying a chosen YUMARC format concept (field, subfield, subsubfield...).

2.1. Window Button Description

This part features the description of the functionality of the buttons present on the windows for YUMARC format concepts modification.

The first set of buttons relates to the navigation through the corresponding data (instances) in the database for each of the chosen concepts. It contains the buttons: **First**, **Previous**, **Next** and **Last**.

The **Preview** button is used for finding the desired instance in the database. After pressing that button, a separate window pops up (different windows for different concepts) where the user chooses values related to the instance being looked for. When this window is opened, it initially contains the values from the window where the preview has been activated.

The next set of buttons deals with actions of entering, modification and deleting instances in the database.

The **New** button enables entering a new instance. After this button is pressed, the user is expected to enter new data and thus form a new instance.

The **Update** button enables the current instance to be modified.

The **Clear** button enables the current instance to be deleted. When deleting, only the **OK** and **Cancel** buttons are available (all others are unavailable).

Entering, modification and deleting require the action to be confirmed or cancelled, which is being done using the next set of buttons – **OK** and **Cancel**.

The **Help** button is intended to display the user the complete documentation regarding using the window. Depending on the window where the help is invoked, the corresponding content is displayed.

The **Exit** button is used to leave the window and return to the starting window.

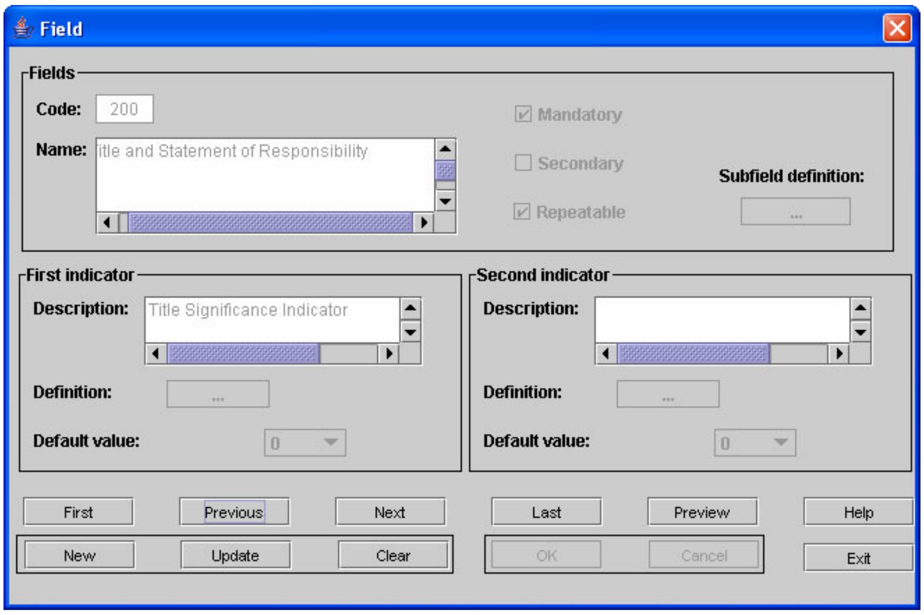
2.2. Window for Fields Modification

The **Field** window, which is shown in Figure 2, is used for entering, modification and deleting fields.

The single-line editor **Code** receives the entry of/displays the field code (3 characters). During entering, when the **OK** button is pressed, the check of the type and length of the field code is preformed. If the field code is not correct, the corresponding message is shown. It is also checked if the entered code is already present in the database. If so, the corresponding message is shown. After the **Update** button is pressed, it is no longer possible to change the field code.

The title of the field at most 160 characters long is entered/displayed in the multi-line editor **Name**. When the **OK** button is pressed during entering or modification, the existence and length of the content of this editor are checked.

The check boxes **Mandatory**, **Repeatable** and **Secondary** are not checked initially (i.e., the field is not mandatory, repeatable and secondary). By checking these check boxes it is achieved that the value in the database is set (i.e., the field is mandatory, repeatable and secondary).



Field

Fields

Code: 200

Name: itle and Statement of Responsibility

☑ Mandatory

☐ Secondary

☑ Repeatable

Subfield definition: ...

First indicator

Description: Title Significance Indicator

Definition: ...

Default value: 0

Second indicator

Description:

Definition: ...

Default value: 0

Figure 2. **Field** window

The **Subfield** definition button is active during entering a new field or updating an existing one. Pressing this button results in opening the window for entering subfields of the corresponding field. This window is the same as the one in Figure 4, except that the code field value is not being chosen, but it is passed from the **Field** window. The user must enter at least one subfield for every new field. When the **OK** button is pressed, the check is performed and the corresponding message is displayed.

The multi-line editor **Description** (of the first indicator) enables entering a description of the first indicator at most 240 characters long. If the description is entered, entering the first indicator value is required. When the **OK** button is pressed, it is checked if indicators for the entered first indicator description are defined. If not, the corresponding message is displayed

The value of the first indicator is defined in another window, which is invoked by pressing the button **Definition** in the part concerning the first indicator. This window is the same as the one shown in Figure 8, except that the value for field code is not being chosen, but it is passed from the **Field** window.

The value of the first indicator is chosen from the predefined set of first indicator values in the drop-down list **Default value**. If the indicator is not defined, it is impossible to access that drop-down list.

The functionality of the part of the window concerning the second indicator is the same as described for the first indicator (except that data actually concern the second indicator).

Figure 2 shows the field *200* whose name is *Title and Statement of Responsibility*. The field is mandatory and secondary. The description of the first indicator of this field is *Title Significance Indicator*, and its default value is *0*.

Window for finding the desired instance of the field concept. This window, which is shown in Figure 3, is invoked by pressing the **Preview** button. The user chooses the field code. After pressing the **OK** button, the desired instance is found (the field *200* in Figure 3). The action is cancelled by pressing the **Cancel** button.

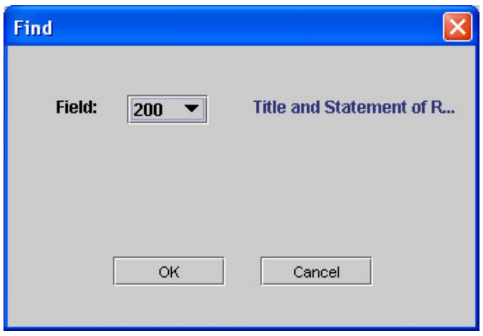


Figure 3. Window for finding a field concept instance

2.3. Window for Subfields Modification

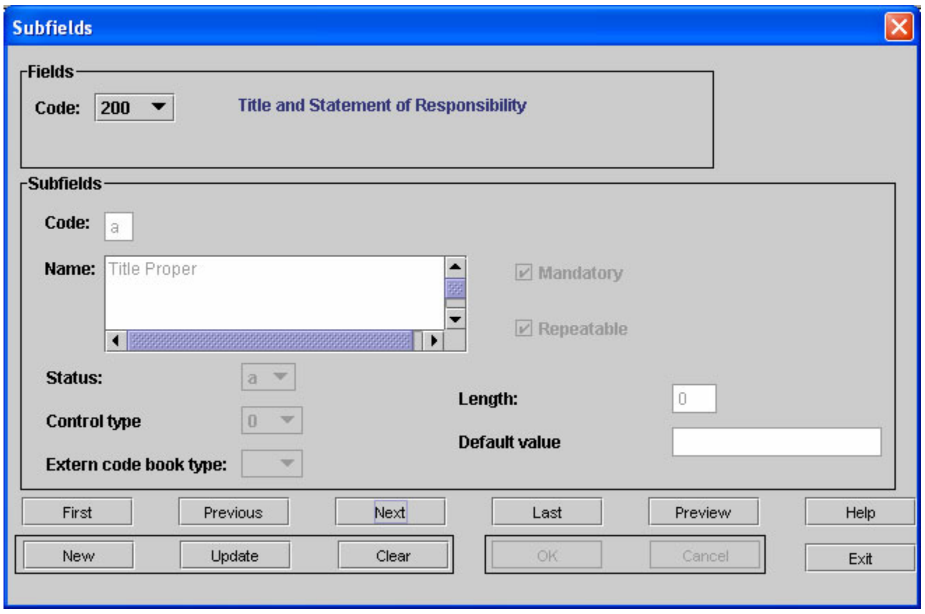
The window **Subfields**, shown in Figure 4, is used for entering, modification and deleting subfields.

The drop-down list (field) **Code** contains all entered field codes which have at least one subfield defined. When choosing the field code, all window components display corresponding values related to the first instance of the subfield of the chosen field.

The single-line editor (field) **Code** receives the entry of/displays the subfield code (1 character). During entering, when the **OK** button is pressed, the check of the type and length of the subfield code is preformed. If the code is not correct, the corresponding message is shown. It is also checked if the entered subfield code, together with the chosen field code, is already present in the database. If so, the corresponding message is shown. After the **Update** button is pressed, it is no longer possible to change the subfield code nor choose the field code.

The name of the field at most 160 characters long is entered/displayed in the multi-line editor **Name**. When the **OK** button is pressed during entering or modification, the existence and length of the content of this editor are checked.

The check boxes **Mandatory**, **Repeatable** and **Secondary** are not checked initially (i.e., the subfield is not mandatory and repeatable). By checking these check boxes it is achieved that the value in the database is set (i.e., the subfield is mandatory and repeatable).


Subfields

Fields

Code: 200 Title and Statement of Responsibility

Subfields

Code: a

Name: Title Proper

☑ Mandatory

☑ Repeatable

Status: a

Control type 0

Extern code book type:

Length: 0

Default value

Figure 4. **Subfields** window

The drop-down list **Status**, has an *a* (active) chosen as a predefined value. The user can choose between the two values of the status: *a* (active) or *p* (passive).

The drop-down list **Control type** contains all entered types of control. The user can choose one of the offered control types. If the chosen control type is different from *1* (the control through the external code book), then the drop-down list **Extern control book type** is not accessible. The predefined value of the control type is *0*.

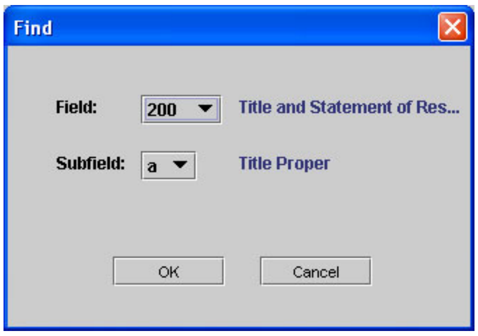
The drop-down list **Extern control book type** contains all entered types of external code books. The user can choose one of the offered types of external code books. If the chosen control type is different from *1*, the drop-down list **Extern control book type** is not accessible.

The single-line editor **Length** receives the entry of/displays the subfield content length (2 digits at most). During entering and modification, when the **OK** button is pressed, the check of the type and length of the content of this editor is preformed.

The single-line editor **Default value** receives the entry of/displays the default value of the subfield at most 15 characters long. During entering and modification, when the **OK** button is pressed, the check of the length of the content of **Default value** editor is done. It must not be bigger than the defined subfield length.

Figure 4 shows the subfield *a* of the field *200*. The name of the *a* subfield is *Title Proper*. The subfield is mandatory and repeatable. The status of the field is *a* (active), while the control type and length are *0*.

Window for finding the desired subfield concept instance. This window, shown in Figure 5, is invoked by pressing the **Preview** button. The user chooses the desired field code in the drop-down list which contains field codes. After having done that, the user chooses the subfield code. The desired instance is found after pressing the **OK** button (in Figure 5 the subfield *a* of the *200* field). The action is cancelled by pressing the **Cancel** button.



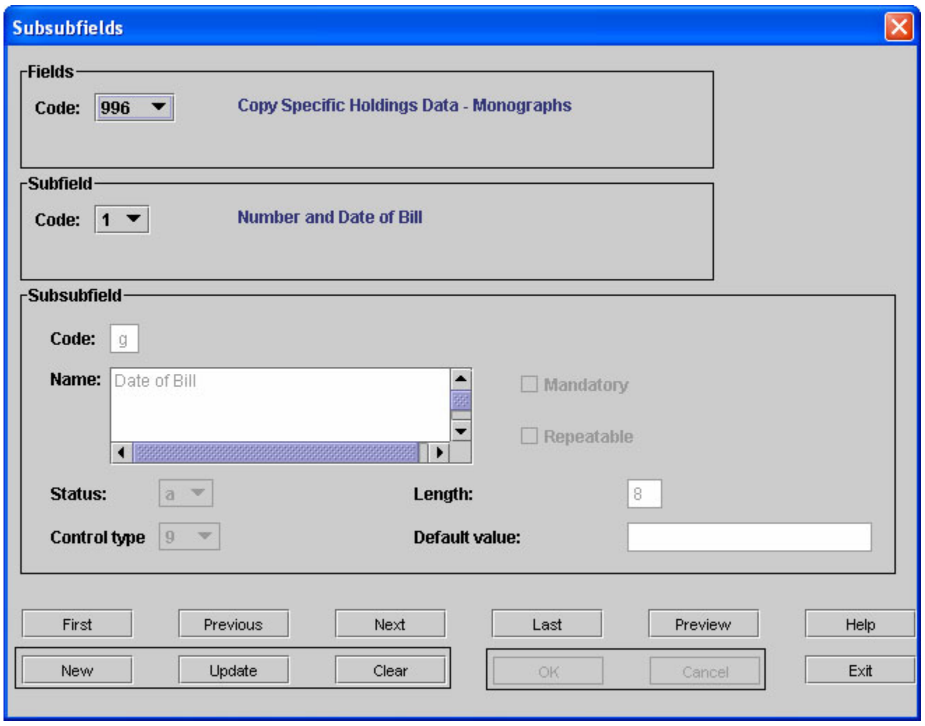
Field: 200 Title and Statement of Res...
Subfield: a Title Proper

Figure 5. Window for finding an instance of the subfield concept

2.4. Window for Subfields Modification

The window **Subfields**, shown in Figure 6, is used for entering, modification and deleting subfields.

The drop-down list (field) **Code** contains all field codes, When the user chooses the field code, whose at least one subfield has at least one subsubfield defined, the remaining window components show corresponding values of the instance whose field code is equal to the chosen field code. Otherwise, the user is automatically transferred to the mode for entering a new subfield instance.


Subsubfields

Fields
Code: 996 Copy Specific Holdings Data - Monographs

Subfield
Code: 1 Number and Date of Bill

Subsubfield
Code: g
Name: Date of Bill
Mandatory
Repeatable
Status: a
Length: 8
Control type 9
Default value:

Figure 6. **Subfileds** window

The drop-down list (subfield) **Code** contains all subfield codes which are related to the chosen field code. When the user chooses a subfield code which, together with the chosen field code, has at least one subsubfield defined, the remaining window components display the corresponding values of the instance whose field code is equal to the chosen field code, and the subfield code is equal to the chosen subfield code. Otherwise, the user is automatically transferred to the mode for entering a new subsubfield instance.

The single-line editor (subsubfield) **Code** receives the entry of/displays the subsubfield code (1 character). During entering, when the **OK** button is pressed, the check of the type and length of the subsubfield code is preformed. If the code is not correct, the corresponding message is shown. It is also checked if the entered subsubfield code, together with the chosen field code and the subfield code, are already present in the database. If the codes already exist, the corresponding message is shown. After the **Update** button is pressed, it is no longer possible to change the subsubfield code nor choose the field code and subfield code.

The name of the subsubfield at most 160 characters long is entered/displayed in the multi-line editor **Name**. When the **OK** button is pressed during entering or modification, the existence and length of the content of this editor are checked.

The check boxes **Mandatory** and **Repeatable** are not checked initially (i.e., the subsubfield is not mandatory or repeatable). By checking these check boxes it is achieved that the value in the database is set (i.e., the subsubfield is mandatory and repeatable).

The drop-down list **Status**, has an *a* (active) chosen as a predefined value. The user can choose between the two values of the status: *a* (active) or *p* (passive).

The drop-down list **Control type** contains all entered types of control. The user can choose one of the offered control types.

The single-line editor **Length** receives the entry of/displays the subsubfield content length (2 digits at most). During entering and modification, when the **OK** button is pressed, the check of the existence and length of the content of this editor is preformed.

The single-line editor **Default value** receives the entry of/displays the default value of the subsubfield at most 15 characters long. During entering and modification, when the **OK** button is pressed, the check of the length of the content of **Default value** editor is done – it must not be bigger than the defined subsubfield length.

Figure 6 shows the subsubfield *g* of the subfield *1* of the field *996*. The name of the subsubfield *g* is *Date of Bill*. The status of the subsubfield is *a* (active), the control type is *9* and the length is *8*.

Window for finding the desired instance of the subsubfield concept. This window, shown in Figure 7, is invoked by pressing the **Preview** button. The user chooses the desired field code and subfield code. After having done that, the user chooses the subsubfield code. The desired instance is found after pressing the **OK** button (in Figure 7 the subsubfield *g* of the subfield *1* of the *996* field). The action is cancelled by pressing the **Cancel** button.

2.5. Window for First Indicator Modification

The window **First indicator**, shown in Figure 8, is used for entering, modification and deleting first indicators.

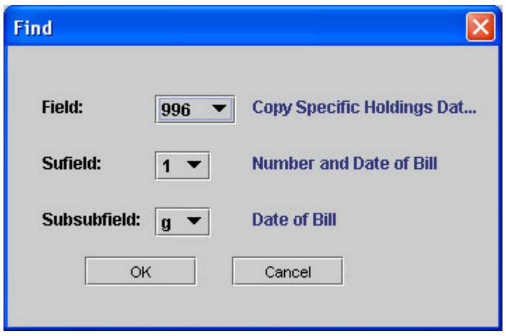


Figure 7. Window for finding an instance of the subsubfield concept

The drop-down list **Code** contains all entered field codes. When the user chooses the field code whose first indicator is defined, the remaining window components show corresponding values of the instance whose field code is equal to the chosen field code. Otherwise, the user is automatically transferred to the mode for entering a new first indicator instance.

The single-line editor **Value** receives the entry of/displays the value of the first indicator (1 digit). During entering, when the **OK** button is pressed, the check of the type and length of the first indicator value is preformed. If the value is not correct, the corresponding message is shown. It is also checked if the value of the first indicator, together with the chosen field code, are already present in the database. If values already exist, the corresponding message is shown. After the **Update** button is pressed, it is no longer possible to change the first indicator value nor choose the field code.

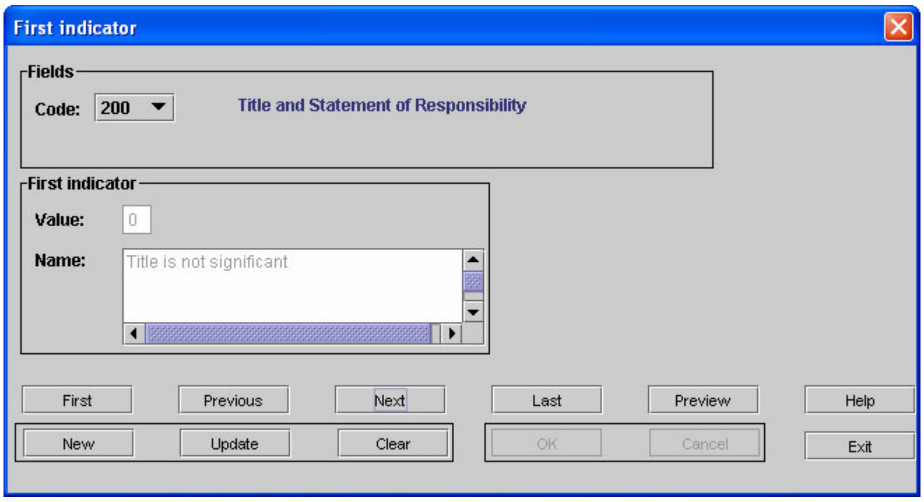
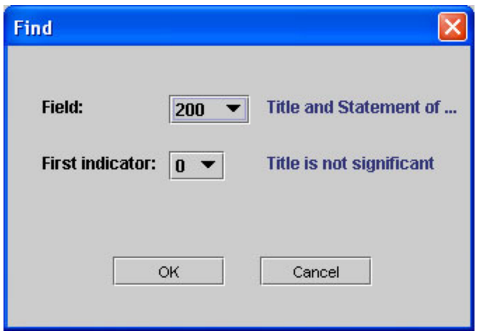


Figure 8. **First indicator** window

The name of the first indicator at most 160 characters long is entered/displayed in the multi-line editor **Name**. When the **OK** button is pressed during entering or modification, the existence and length of the content of this editor are checked.

Figure 8 shows the field *200* with the first indicator value *0*, whose name is *Title is not significant*.

Window for finding the desired instance of the first indicator concept. This window, shown in Figure 9, is invoked by pressing the **Preview** button. The user chooses the desired field code from the list containing field codes. After having done that, the user chooses the first indicator value. The desired instance is found after pressing the **OK** button (in Figure 9 the first indicator value *0* of the *200* field). The action is cancelled by pressing the **Cancel** button.


Find

Field: 200 Title and Statement of ...

First indicator: 0 Title is not significant

Figure 9. Window for finding an instance of the first indicator concept

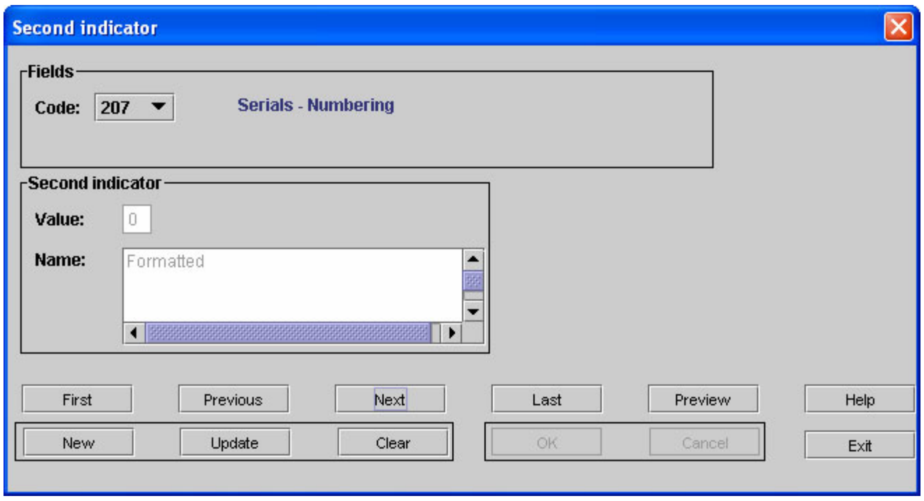
2.6. Window for Second Indicator Modification

The window **Second indicator**, shown in Figure 10, is used for entering, modification and deleting second indicators.

The drop-down list **Code** contains all entered field codes. When the user chooses the field code whose second indicator is defined, the remaining window components show corresponding values of the instance whose field code is equal to the chosen field code. Otherwise, the user is automatically transferred to the mode for entering a new second indicator instance.

Functionality of the components **Value** and **Name** in the **Second indicator** window is the same as that of the components **Value** and **Name** in the **First indicator** window (except that data concerns the second indicator).

Figure 10 shows the field *207* with the second indicator value *0*, whose name is *Formatted*.


Second indicator

Fields

Code: 207 Serials - Numbering

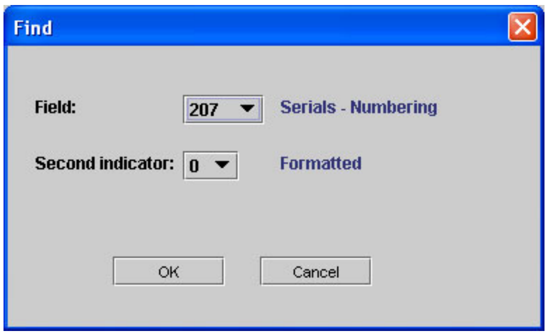
Second indicator

Value: 0

Name: Formatted

Figure 10. **Second indicator** window

Window for finding the desired instance of the second indicator concept. This window, shown in Figure 11, is invoked by pressing the **Preview** button. The user chooses the desired field code from the list containing field codes. After having done that, the user chooses the second indicator value. The desired instance is found after pressing the **OK** button (in Figure 11, the second indicator value *0* of the *207* field). The action is cancelled by pressing the **Cancel** button.


Find

Field: 207 Serials - Numbering

Second indicator: 0 Formatted

Figure 11. Window for finding an instance of the second indicator concept

2.7. Window for Type of External Code Book Modification

The window **External code book type**, shown in Figure 12, is used for entering, modification and deleting the external code book type.

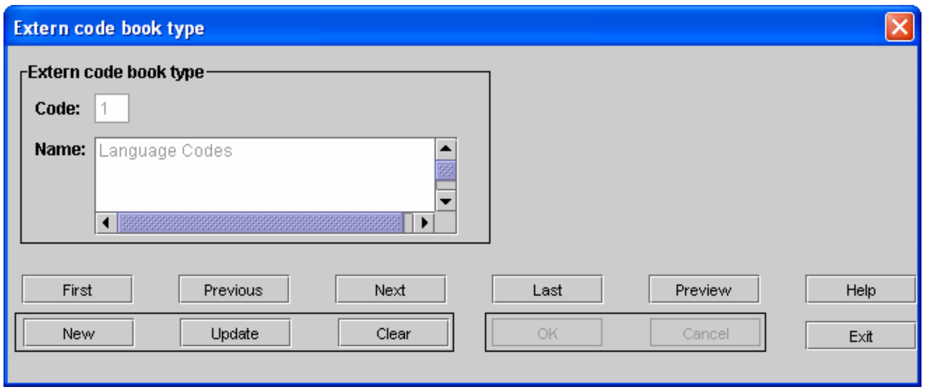


Figure 12. External code book type window

The single-line editor **Code** receives the entry of/displays the code of the external code book type (at most 2 digits). During entering, when the **OK** button is pressed, the check of the type and length of the external code book code is preformed. If the code is not correct, the corresponding message is shown. It is also checked if the entered code is already present in the database. If the code already exists in the database, the corresponding message is shown. After the **Update** button is pressed, it is no longer possible to change the code of the external code book.

The name of the external code book type at most 160 characters long is entered/displayed in the multi-line editor **Name**. When the **OK** button is pressed during entering or modification, the existence and length of the content of this editor are checked.

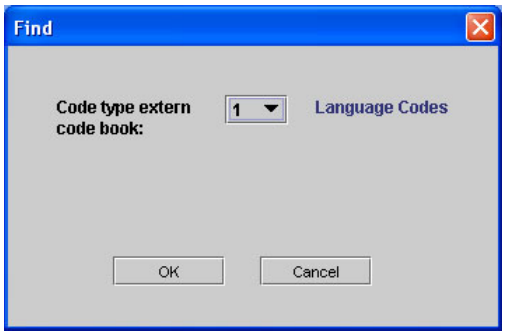
Figure 12 shows the type of the external code book *1*, whose name is *Language codes*.

Window for finding the desired instance of the external code book type. This window, shown in Figure 13, is invoked by pressing the **Preview** button. The user chooses the desired external code book type code. The desired instance is found after pressing the **OK** button (in Figure 13, the external code book type). The action is cancelled by pressing the **Cancel** button.

2.8. Window for External Code Book Modification

The window **External code book**, shown in Figure 14, is used for entering, modification and deleting the external code book.

The drop-down list (external code book type) **Code** contains all entered external code book type codes. When the user chooses the external code book type code which has at least one external code book code associated to it, the remaining window components show corresponding values of the instance whose external code book type code is equal to the chosen external code book type code. Otherwise, the user is automatically transferred to the mode for entering a new external code book instance.


Find

Code type extern code book: 1 Language Codes

Figure 13. Window for finding the instance of the concept of the external code book type

The single-line editor (external code book) **Code** receives the entry of/displays the code of the external code book type (at most 15 characters). During entering, when the **OK** button is pressed, the check of the type and length of the external code book code is preformed. If the code is not correct, the corresponding message is shown. It is also checked if the entered external code book code, together with the chosen external code book type code, are already present in the database. If the codes already exist in the database, the corresponding message is shown. After the **Update** button is pressed, it is no longer possible to change the code of the external code book nor choose the external code book type code.

The name of the external code book at most 160 characters long is entered/displayed in the multi-line editor **Name**. When the **OK** button is pressed during entering or modification, the existence and length of the content of this editor are checked.

Figure 14 shows the type of the external code book *1* (*Language codes*) with the external code book code *ace*, whose name is *Achinese*.

Window for finding the desired instance of the external code book. This window, shown in Figure 15, is invoked by pressing the **Preview** button. The user chooses the desired external code book type code from the drop-down list containing external code book type codes. After having chosen the external code book type code, the user chooses the external code book code. The desired instance is found after pressing the **OK** button (in Figure 15, the external code book *scc* of the external code book type *1*). The action is cancelled by pressing the **Cancel** button.

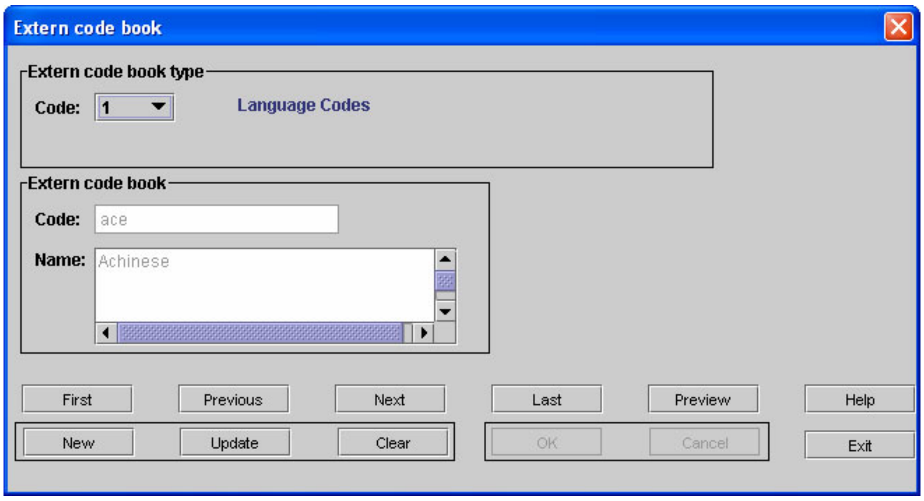


Figure 14. **External code book** window

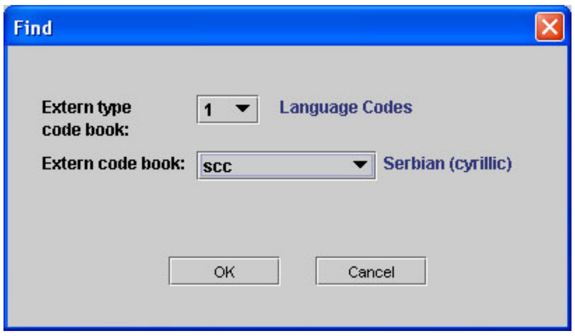


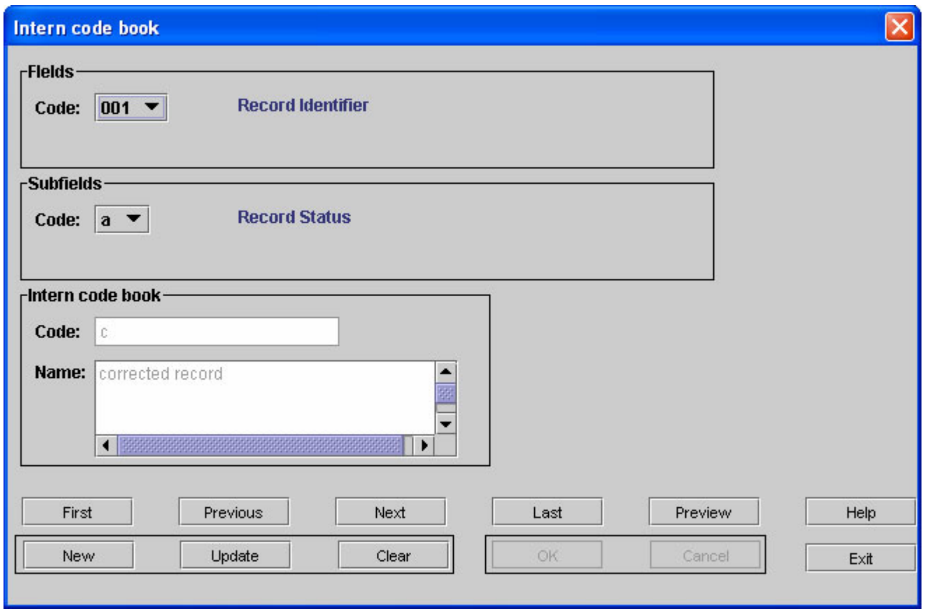
Figure 15. Window for finding an instance of the external code book concept

2.9. Window fop Internal Code Book

The window **Internal code book**, shown in Figure 16, is used for entering, modification and deleting the internal code book.

The drop-down list (field) **Code** contains all entered field codes, When the user chooses the field code whose at least one subfield has associated at least one internal code book code, the remaining window components show corresponding values of the instance whose field code is equal to the chosen field code. Otherwise, the user is automatically transferred to the mode for entering a new internal code book instance.

The drop-down list (subfield) **Code** contains all entered subfield codes related to the chosen field code. When the user chooses the subfield code which, together with chosen field code, has at least one internal code book code associated to it, the remaining window components show corresponding values of the instance whose field code is equal to the chosen field code, and the subfield code is equal to the chosen subfield code. Otherwise, the user is automatically transferred to the mode for entering a new internal code book instance. In the process of entering a new instance, when the **OK** button is pressed, if the control type for the chosen subfield is not 2 (control through the internal code book – for subfields), the corresponding message is displayed and the control type for that field is set to 2.


Intern code book

Fields
Code: 001 Record Identifier

Subfields
Code: a Record Status

Intern code book
Code: c
Name: corrected record

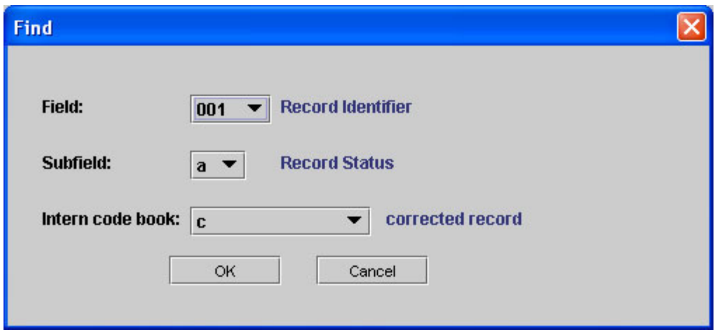
Figure 16. **Internal code book** window

The single-line editor (internal code book) **Code** receives the entry of/displays the code of the internal code book code (at most 15 characters). During entering, when the **OK** button is pressed, the check of the type and length of the internal code book code is preformed. If the code is not correct, the corresponding message is shown. It is also checked if the entered internal code book code, the chosen field and subfield codes are already present in the database. If the codes already exist in the database, the corresponding message is shown. After the **Update** button is pressed, it is no longer possible to change the code of the internal code book nor choose the field and subfield codes.

The name of the internal code book at most 160 characters long is entered/displayed in the multi-line editor **Name**. When the **OK** button is pressed during entering or modification, the existence and length of the content of this editor are checked.

Figure 16 shows the code of the internal code book *c* of the subfield *a* of the field *001*. The name of the internal code book *c* is *corrected record*.

Window for finding the desired instance of the internal code book concept. This window, shown in Figure 17, is invoked by pressing the **Preview** button. The user chooses the desired field and subfield codes. After having chosen them, the user chooses the internal code book code. The desired instance is found after pressing the **OK** button (in Figure 17, the internal code book code *c* of the subfield *a* of the field *001*). The action is cancelled by pressing the **Cancel** button.


Find

Field: 001 Record Identifier

Subfield: a Record Status

Intern code book: c corrected record

Figure 17. Window for finding an instance of the internal code book concept

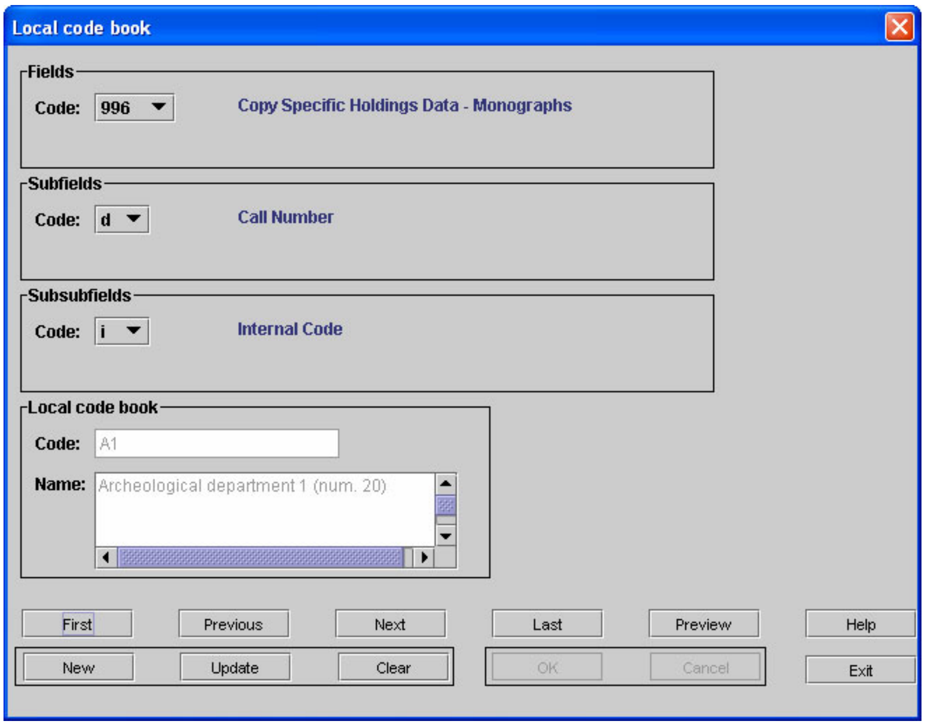
2.10. Window for Local Code Book Modification

The window **Local code book**, shown in Figure 18, is used for entering, modification and deleting the local code book (the subsubfield code book).

The drop-down list (field) **Code** contains all entered field codes whose at least one subfield contains subsubfields. When the user chooses the field code whose at least one subfield and subsubfield have associated at least one local code book code, the remaining window components show corresponding values of the instance whose field code is equal to the chosen field code. Otherwise, the user is automatically transferred to the mode for entering a new local code book instance.

The drop-down list (subfield) **Code** contains all entered subfield codes related to the chosen field code and have subfields. When the user chooses the subfield code which, together with chosen field code, has at least one local code book code associated to it, the remaining window components show the corresponding values of the instance whose field code is equal to the chosen field code, and the subfield code is equal to the chosen subfield code. Otherwise, the user is automatically transferred to the mode for entering a new local code book instance.

The drop-down list (subsubfield) **Code** contains all entered subsubfield codes related to the chosen subfield and field codes. When the user chooses the subsubfield code which, together with chosen field and subfield codes, has at least one local code book code associated to it, the remaining window components show the corresponding values of the instance whose field code is equal to the chosen field code, the subfield code is equal to the chosen subfield code, and the subsubfield code is equal to the chosen subsubfield code. Otherwise, the user is automatically transferred to the mode for entering a new local code book instance. In the process of entering a new instance, when the **OK** button is pressed, if the control type for the chosen subsubfield is not 2 (control through the local code book – for subsubfields), the corresponding message is displayed and the control type for that field is set to 2.


Local code book

Fields
Code: 996 Copy Specific Holdings Data - Monographs

Subfields
Code: d Call Number

Subsubfields
Code: i Internal Code

Local code book
Code: A1
Name: Archeological department 1 (num. 20)

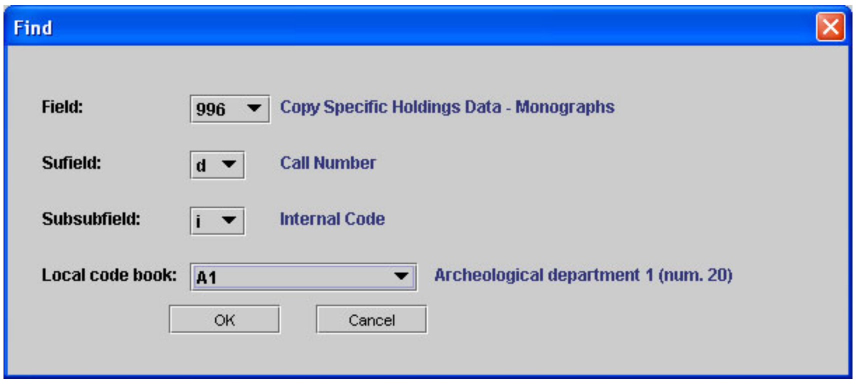
Figure 18. **Local code book** window

The single-line editor (local code book) **Code** receives the entry of/displays the code of the local code book code (at most 15 characters). During entering, when the **OK** button is pressed, the check of the type and length of the local code book code is preformed. If the code is not correct, the corresponding message is shown. It is also checked if the entered local code book code, the chosen field and subfield and subsubfield codes are already present in the database. If the codes already exist in the database, the corresponding message is shown. After the **Update** button is pressed, it is no longer possible to change the code of the local code book nor choose the field and subfield and subsubfield codes.

The name of the local code book at most 160 characters long is entered/displayed in the multi-line editor **Name**. When the **OK** button is pressed during entering or modification, the existence and length of the content of this editor are checked.

Figure 18 shows the local code book code *A1*, whose name is *Archeological department 1 (num. 20)*. The code of the local code book *A1* belongs to the subsubfield *i* of the subfield *d* of the field *996*.

Window for finding the desired instance of the local code book concept. This window, shown in Figure 19, is invoked by pressing the **Preview** button. The user chooses the desired field and subfield and subsubfield codes. After having chosen these codes, the user chooses the local code book code. The desired instance is found after pressing the **OK** button (in Figure 19, the local code book code *A1* of the subsubfield *l* of the subfield *d* of the field *996*). The action is cancelled by pressing the **Cancel** button.


Find

Field: 996 Copy Specific Holdings Data - Monographs

Sufield: d Call Number

Subsubfield: i Internal Code

Local code book: A1 Archeological department 1 (num. 20)

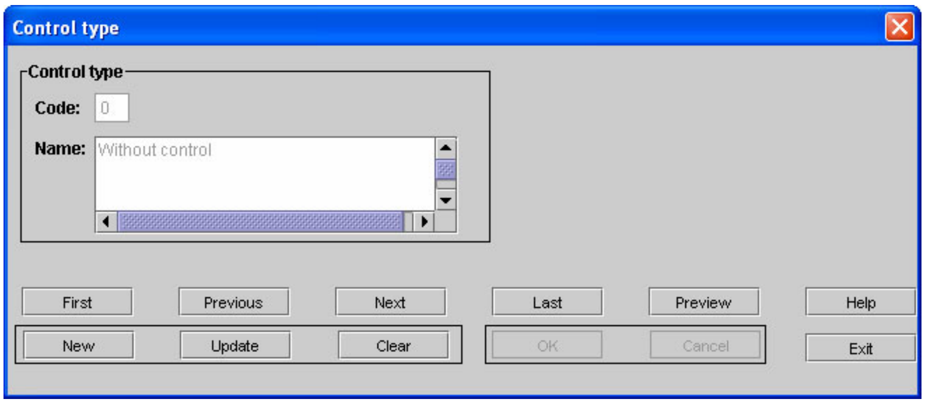
Figure 19. Window for finding an instance of the local code book concept

2.11. Window for Control Type Modification

The window **Control type**, shown in Figure 20, is used for entering, modification and deleting the control type.

The single-line editor **Code** receives the entry of/displays the code of the control type (at most 2 digits). During entering, when the **OK** button is pressed, the check of the type and length of the control type code is preformed. If the code is not correct, the corresponding message is shown. It is also checked if the entered control type code is already present in the database. If the code already exists in the database, the corresponding message is shown. After the **Update** button is pressed, it is no longer possible to change the code of the control type.

The name of the control type at most 160 characters long is entered/displayed in the multi-line editor **Name**. When the **OK** button is pressed during entering or modification, the existence and length of the content of this editor are checked.



Code: 0

Name: Without control

Figure 20. **Control type** dialog

Figure 20 shows the control type code *0*, whose name is *Without control*.

Window for finding the desired instance of the control type concept. This window, shown in Figure 21, is invoked by pressing the **Preview** button. The user chooses the desired control type code. The desired instance is found after pressing the **OK** button (in Figure 21 the control type code 0). The action is cancelled by pressing the **Cancel** button.

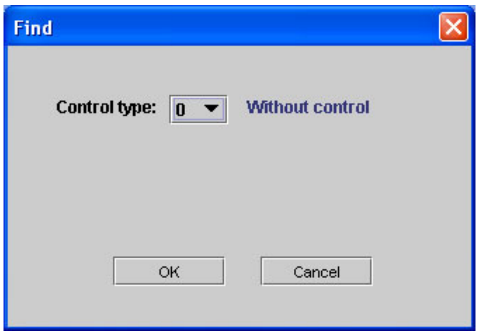


Figure 21. Window for finding an instance of control type concept

3. Conclusion

The library software system BISIS ver. 3.0, which is implemented in the Java environment, is a system that supports library operation. The system is based on the YUMARC format, and as its integral part the application enabling all YUMARC format concepts has been implemented. The configurability of BISIS software system regarding definition and maintenance of the YUMARC format to meet the needs of any particular library has been provided in this way.

References

[1] *MARC Format Overview*. Network Development and MARC Standards Office, Library of Congress, 08/13/2003., http:// www.loc.gov/marc/status.html

[2] *UNIMARC Manual: Bibliographic Format/International Federation of Library Associations and Institutions*, IFLA Universal Bibliographic Control and International MARC Programme, New Providence, London 1994.

[3] Budimir, G., *Marc records and XML*, INFOteka, February 2004, year. 4, num. 1, Belgrade (in print)

[4] Surla D., Konjović Z., et all, *Instruction Manual for Library Software System BISIS version 3*, Group for Information Technologies, Novi Sad, 2003. (in Serbian)

[5] Holo, I., Pupovac, B., Tošić, T., *The user interface for data maintenance of the UNIMARC format*, Proceedings of the XIII Conference on Applied Mathematics PRIM'98, Igalo, 1998., pp. 43-50

[6] Tričković, I., Holo, I., Pupovac, B., *Data model of the UNIMARC format in Oracle Designer/2000*, Proceedings of the XIII Conference on Applied Mathematics PRIM'98, Igalo, 1998., pp. 179-186.

[7] Eckel, B., *Thinking in Java*, 2nd edition, 2000., http://www.EckelObjects.com

Maintenance of the Librarian Working Environment

Biljana Pupovac
Faculty of Science and Mathematics, University of Novi Sad
biljanap@uns.ns.ac.yu

Abstract. The librarian working environment enables defining various levels and modes of processing publications. In addition to this, duties and responsibilities during publications processing are defined for every librarian. This paper presents a detailed description of the use of the application for Librarian working envronment modification, which is a constituent part of the BISIS software system.

1. Introduction

In order to provide the complete functionality of the BISIS software system, the data concerning librarian working environment are stored in the Librarian working environment database (Working environment database), which is an integral part of the system. These data are used with the purpose of adapting the application for bibliographical processing to particular requirements of every individual librarian, as well as to document types being processed by him/her. The model of the Working environment database is given in papers [1, 2, 3].

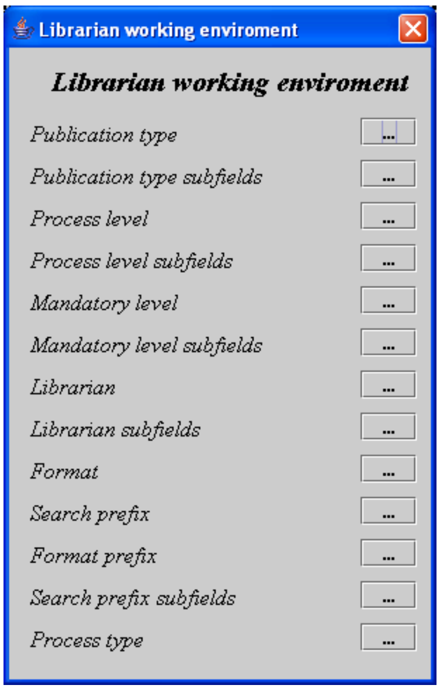
The data about a librarian must be present in the Working environment database, so as to enable him/her to work with the application for bibliographical material processing. When the application is started, the data from the database are being passed and the working environment is being adjusted for the logged-on librarian. Adapting the working environment consists of setting the publication type, processing level, mandatory level, setting the five search prefixes and the format of display for every logged-on librarian separately.

The Working Environment database also facilitates the work of a librarian with different document types. Monongraphs, serials, Ph.D. theses are examples of document types. For each document type fields/subfields enabling processing data about such a document can be defined. Depending on the type and scope of data being defined when processing a document, there are several processing levels for every publication type. For each processing level the fields/subfields for which values are entered at that level are defined. Each level of processing defined for a particular publication type can be associated with several mandatory levels. A document is considered processed if values for fields/subfields defined at that mandatory level are entered.

The Working environment database also contains different display formats which account for different report and search results displays. In addition to this, this database features search prefixes which are used when forming quearies on the Bibliographical material database.

2. Application for Librarian Working Environment Modification

The application for Librarian working environment modification has been implemented within the software BISIS ver.3.0. It has been developed in the Java environment [4, 5].


Librarian working enviroment

Librarian working enviroment

Publication type ...
Publication type subfields ...
Process level ...
Process level subfields ...
Mandatory level ...
Mandatory level subfields ...
Librarian ...
Librarian subfields ...
Format ...
Search prefix ...
Format prefix ...
Search prefix subfields ...
Process type ...

Figure 1. Starting window

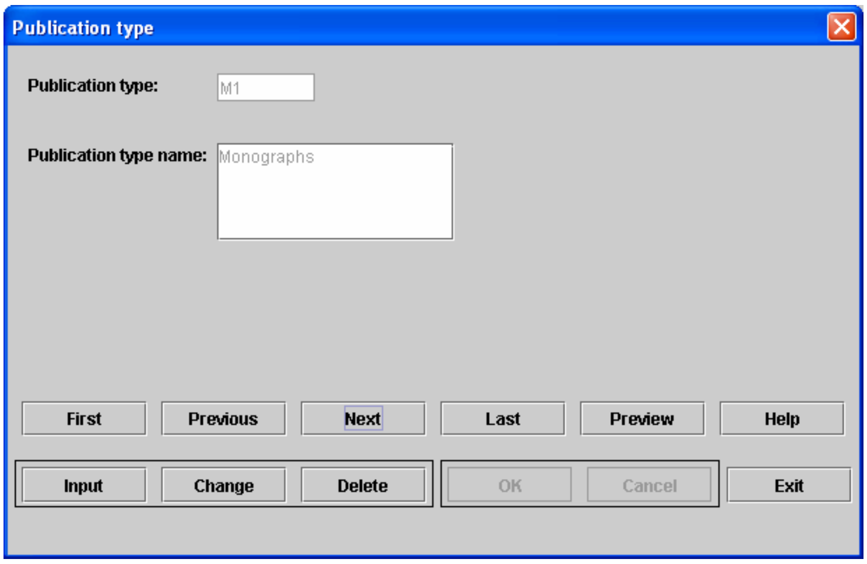
The appearance of the starting window of the librarian working environment modification is shown in Figure 1. This window enables transition to the concrete window for the modification of the chosen concept of the Librarian working environment (publication type, publication type subfields, processing level...)

2.1. Description of Window Buttons

The description of window buttons functionality enabling the Librarian working environment modification is the same as the one for the YUMARC format modification (presented in the paper *Maintenance of the YUMARC format* [6])

2.2. Window for Publication Type Modification

The window **Publication type**, which is shown in Figure 2, is used for entering, modification and deleting the publication type.



Publication type: M1

Publication type name: Monographs

Figure 2. **Publication type** window

The single-line editor **Publication type** receives the entry of/displays the value of the publication type (2 characters). During entering, when the **OK** button is pressed, the value type and length of the publication type are checked. If these values are not correct, the corresponding message is shown. It is also checked if the entered value is already present in the database. If so, the corresponding message is displayed. After the button **Change** is pressed, the publication type value can no longer be changed.

The multi-line editor **Publication type name** receives the entry of/displays the name of the publication type at most 160 characters long. When the button **OK** is pressed during entering and modification, the existence and length of the editor's content is checked.

Figure 2 shows the *M1* publication type, with the name *Monographs*.

The window for finding a desired instance of the publication type concept. This window, which is given in Figure 3, is invoked by pressing the button **Preview**. The user then chooses the publication type value. By pressing the **OK** button, the desired instance is found (the publication type *M1* in Figure 3). Should the **Cancel** button be pressed, this action is cancelled.

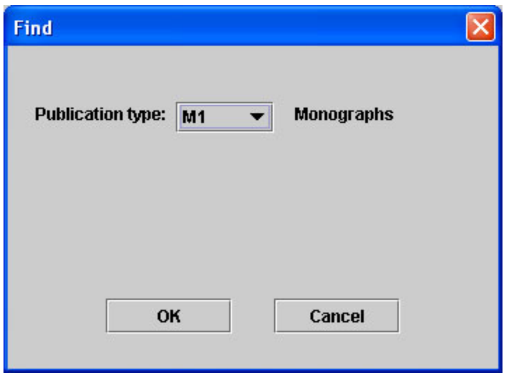


Figure 3. Window for finding the instance of the publication type concept

2.3. Window for Publication Type Subfields Modification

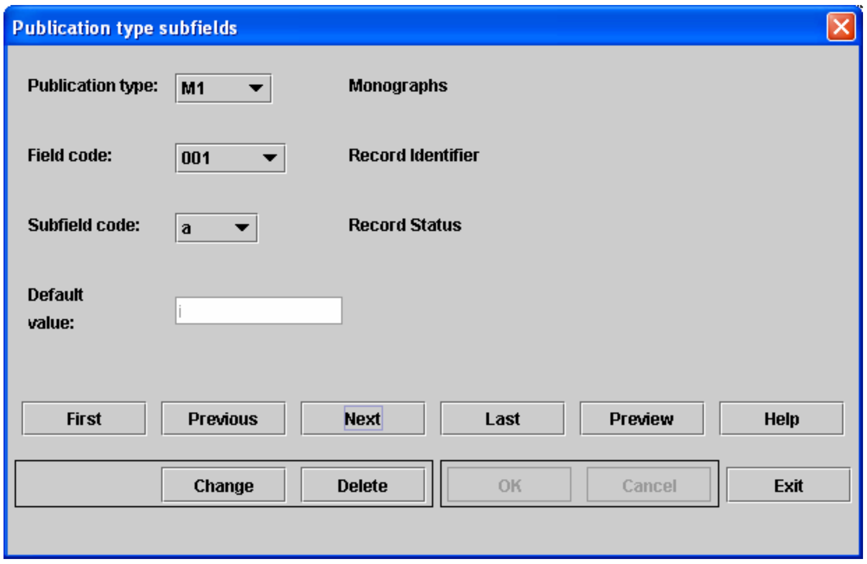
The window **Publication type subfields**, which is shown in Figure 4, is used for entering, modification and deleting publication type subfields.

The drop-down list **Publication type** contains all entered values of the publication type. When the user chooses a publication type value for which publication type subfields are already defined, the remaining window components display corresponding values of the instance whose publication type value is equal to the chosen one. Otherwise, the user is automatically transferred to the mode for entering a new instance of publication type subfields.

The drop-down list **Field code** contains all entered field codes. When the user chooses a field code which is already associated to the chosen publication type value, together with any of its subfields, the remaining window components display the corresponding values of the instance whose publication type value is equal to the chosen field code. Otherwise, the user is automatically transferred to the mode for entering a new instance of publication type subfields.

The drop-down list **Subfield code** contains all subfield codes related to the chosen field code. When the user chooses a subfield code which is, together with the chosen field code, already associated to the chosen publication type, the single-line editor containing the default subfield value displays the corresponding value of the instance whose publication type value is equal to the chosen publication type value, the field code equal to the chosen field code and the subfield code equal to the chosen subfield code. Otherwise, the user is automatically transferred to the mode for entering a new instance of publication type subfields.

The **Default value** receives the entry of/displays the default subfield value in the single-line editor. If the length of the selected subfield has been defined (during entering the subfield), when entering or modification are performed, the length of the entered default value is checked on pressing the **OK** button.


Publication type subfields

Publication type: M1 Monographs

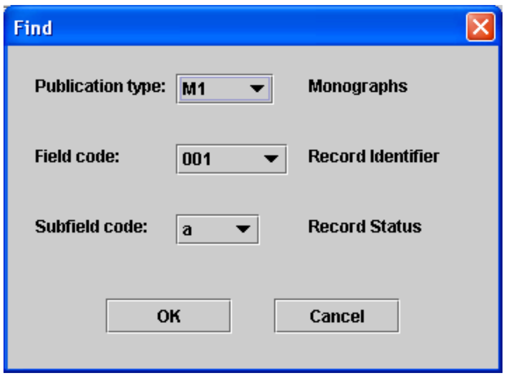
Field code: 001 Record Identifier

Subfield code: a Record Status

Default value: i

Figure 4. Publication type subfields window

Figure 4 shows the default value *i* of the subfield *a* of the field *001*, which is associated to the *M1* publication type.


Find

Publication type: M1 Monographs

Field code: 001 Record Identifier

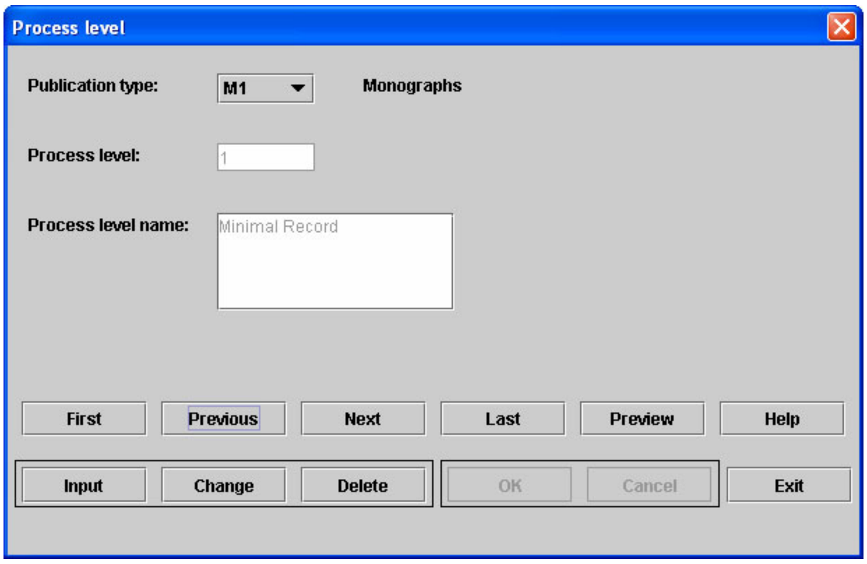
Subfield code: a Record Status

Figure 5. Window for finding an instance of the publication type subfield concept

Window for finding the desired instance of the publication type subfield concept. This window, shown in Figure 5, is invoked by pressing the **Preview** button. The user chooses the desired publication type. After it has been chosen, the user chooses the field and subfield codes. The desired instance is found by pressing the **OK** button (the subfield code *a* of the field *001* associated to the *M1* publication type in Figure 5). The action is cancelled by pressing the **Cancel** button.

2.4. Window for Processing Level Modification

The window **Processing level**, shown in Figure 6, is used for entering, modification and deleting the processing level.



Publication type: M1 Monographs

Process level: 1

Process level name: Minimal Record

Figure 6. **Processing level** window

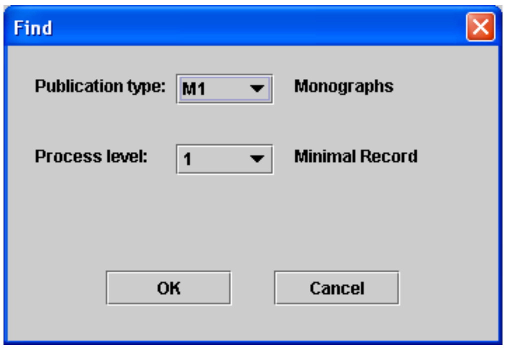
The drop-down list **Publication type** contains all values of the publication type. When the user chooses the publication type whose processing level is already defined, the remaining window components display the corresponding values of the instance whose publication type value is equal to the chosen one. Otherwise, the user is automatically transferred to the mode for entering a new instance of the processing level.

The single-line editor **Processing level** receives the entry of/displays the value of the processing level (at most 2 characters). During entering, when the **OK** button is pressed, the check of the type and length of the processing level value is preformed. If the value is not correct, the corresponding message is shown. It is also checked if the entered level of processing level and the chosen publication type are already present in the database. If so, the corresponding message is shown. After the **Change** button is pressed, it is no longer possible to change the processing level value nor choose the publication type value.

The name of the processing level long at most 160 characters is entered/displayed in the multi-line editor **Processing level name**. When the **OK** button is pressed during entering or modification, the existence and length of the content of this editor are checked.

Figure 6 shows the title of processing level named *Minimal record* associated to the publication type *M1* with the corresponding processing level *1*.

The window for finding the desired instance of the processing level concept. This window, shown in Figure 7, is invoked by pressing the **Preview** button. The user chooses the desired publication type. After having chosen the publication type value, the user chooses the processing level value. By pressing the **OK** button the desired instance is found (the processing level *1* associated to the publication type *M1*, as shown in Figure 7). When the **Cancel** button is pressed, the action is cancelled.


Find

Publication type: M1 Monographs

Process level: 1 Minimal Record

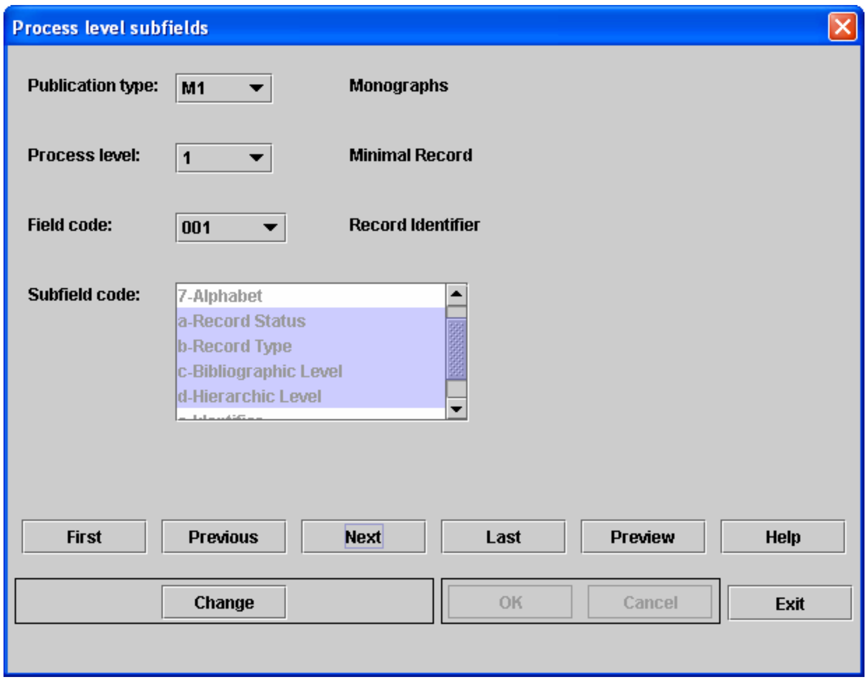
Figure 7. Window for finding an instance of the processing level concept

2.5. Window for Processing Level Subfields Modification

The window **Processing level subfields**, which is shown in Figure 8, is used for defining the processing level subfields.

The drop-down list **Publication type** contains publication type values whose publication type subfields are defined. When the user chooses the publication type value which has processing level subfields defined for at least one processing level, the remaining window components display the corresponding values of the instance whose publication type value is equal to the chosen publication type value.

Otherwise, the user is automatically transferred to the mode for defining the processing level subfields.


Process level subfields

Publication type: M1 Monographs

Process level: 1 Minimal Record

Field code: 001 Record Identifier

Subfield code:
7-Alphabet
a-Record Status
b-Record Type
c-Bibliographic Level
d-Hierarchic Level

Figure 8. Processing level subfields window

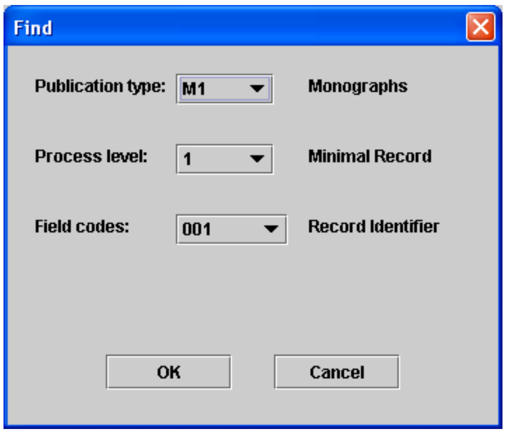
The drop-down list **Processing level** contains all entered processing level values which belong to the chosen publication type. When the user chooses the processing level value which, together with the chosen publication type value, already has defined processing level subfields, the remaining window components display the corresponding values of the instance whose publication type value is equal to the chosen publication type value, and the processing level value is equal to the chosen processing level value. Otherwise, the user is automatically transferred to the mode for defining the processing level subfields.

The drop-down list **Field code** contains all entered field codes. When the user chooses the field code which is already associated to the chosen publication type and processing level values with any of its subfields, the list displaying subfield codes will contain all the subfield codes for the chosen field, with subfields already included into the processing level subfields being marked. Otherwise, the user is automatically transferred to the mode for defining the processing level subfields (in the list displaying subfield codes there are all subfield codes of the chosen field, but they are not marked).

The list **Subfield code** contains all subfield codes of the chosen field, as well as their names. Only the subfields and their names which, together with the chosen publication type values, processing level values and the field code, are already included in the processing level subfields are marked. While defining processing level subfields, when the **OK** button is pressed, it is checked if any of the subfield codes is marked. If neither of them is marked, the corresponding message is shown.

The subfield codes of the field *001* associated to the processing level *1*of the publication type *M1* are shown in Figure 8.

The window for finding the desired instance of the processing level subfields concept. This window, which is given in Figure 9, is invoked by pressing the **Preview** button. The user chooses the publication type value. After it is chosen, the user chooses the processing level value and the field code. The desired instance is chosen after pressing the **OK** button (the field code *001* associated to the processing level *1* of the *M1* publication type, shown in Figure 9). The action is cancelled by pressing the **Cancel** button.


Find

Publication type: M1 Monographs

Process level: 1 Minimal Record

Field codes: 001 Record Identifier

Figure 9. Window for finding an instance of the processing level subfields concept

2.6. Window for Mandatory Level Modification

The window **Mandatory level**, shown in Figure 10, is used for entering, modification and deleting the mandatory level.

The drop-down list **Publication type** contains the publication type values for which at least one mandatory level is defined. When the user chooses a publication type value whose at least one associated processing level has a defined mandatory level, the remaining window components show the corresponding values of the instance whose publication type value is equal to the chosen publication type value.

Otherwise, the user is automatically transferred to the mode for entering a new instance of the mandatory level.

The drop-down list **Processing level** contains all entered processing level values which belong to the chosen publication type. When the user chooses the processing level value which, together with the chosen publication type value, already has a defined mandatory level, the remaining window components display the corresponding values of the instance whose publication type value is equal to the chosen publication type value, and the processing level value is equal to the chosen processing level value. Otherwise, the user is automatically transferred to the mode for entering a new instance of mandatory level.

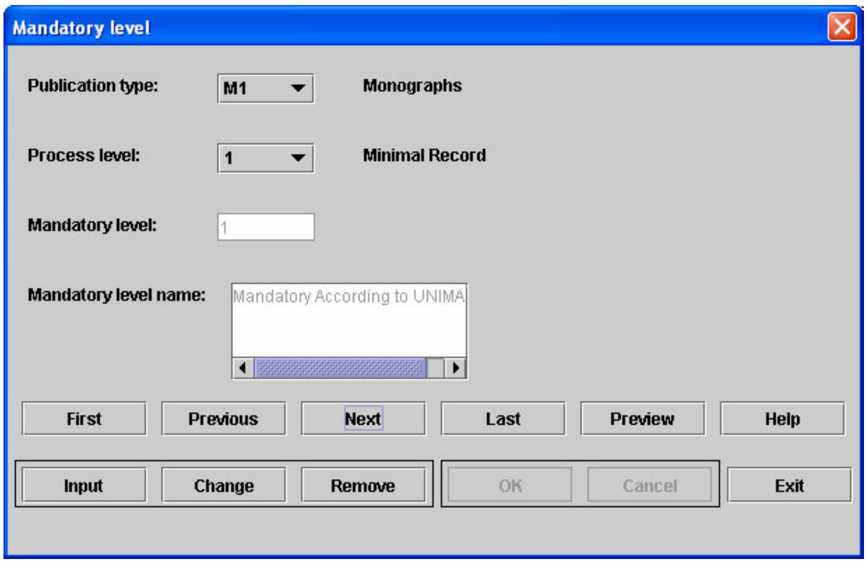

Publication type: M1 Monographs
Process level: 1 Minimal Record
Mandatory level: 1
Mandatory level name: Mandatory According to UNIMA

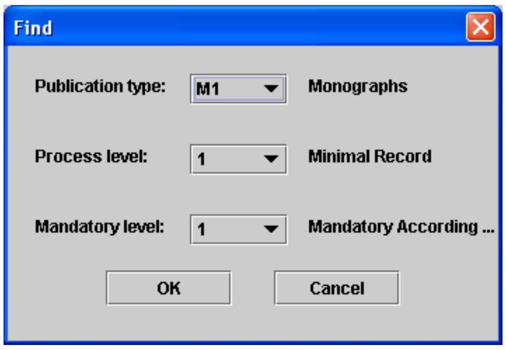
Figure 10. **Mandatory level** window

The single-line editor **Mandatory level** receives the entry of/displays the value of the mandatory level (at most 2 digits). During entering, when the **OK** button is pressed, the check of the type and length of the mandatory level value is preformed. If the value is not correct, the corresponding message is shown. It is also checked if the entered value of the mandatory level, the chosen publication type and processing level value are already present in the database. If so, the corresponding message is shown. After the **Change** button is pressed, it is no longer possible to change the mandatory level value nor choose the publication type or processing level values.

The **Mandatory level name** receives the entry of/displays the name of the mandatory level at most 160 characters long in the multi-line editor. When the button **OK** is pressed during entering and modification, the existence and length of the editor's content is checked.

Figure 10 represents the mandatory level *1* (whose title is *Mandatory According to UNIMARC*) associated to the processing level *1* of the *M1* publication type.

The window for finding the desired instance of the mandatory level concept. This window, shown in Figure 11, is invoked by pressing the **Preview** button. The user chooses the desired publication type and processing level values. After having chosen the publication type and processing level values, the user chooses the mandatory level value. By pressing the **OK** button the desired instance is found (the mandatory level *1* associated to the mandatory level *1* of the publication type *M1*, as shown in Figure 11). When the **Cancel** button is pressed, the action is cancelled.


Find

Publication type: M1 Monographs

Process level: 1 Minimal Record

Mandatory level: 1 Mandatory According ...

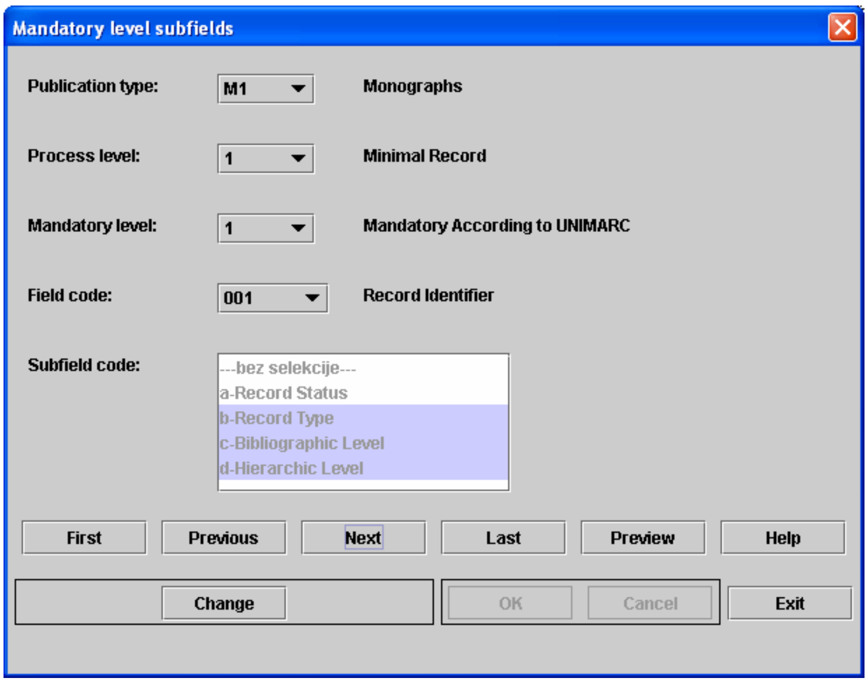
Figure 11. Window for finding an instance of the mandatory level concept

2.7. Window for Mandatory Level Subfields Modification

The window **Mandatory level subfields**, which is shown in Figure 12, is used for defining the mandatory level subfields.

The drop-down list **Publication type** contains the publication type values whose processing level subfields are defined. When the user chooses the publication type value which has mandatory level subfields defined for at least one processing level and mandatory level, the remaining window components display the corresponding values of the instance whose publication type value is equal to the chosen publication type value. Otherwise, the user is automatically transferred to the mode for defining mandatory level subfields.

The drop-down list **Processing level** contains all entered processing level values which belong to the chosen publication type, having processing level subfields defined. When the user chooses the desired processing level value which, for at least one associated mandatory level and the chosen publication type value already has defined mandatory level subfields, the remaining window components display the corresponding values of the instance whose publication type value is equal to the chosen publication type value, and the processing level value is equal to the chosen processing level value. Otherwise, the user is automatically transferred to the mode for defining mandatory level subfields.



Publication type: M1 Monographs

Process level: 1 Minimal Record

Mandatory level: 1 Mandatory According to UNIMARC

Field code: 001 Record Identifier

Subfield code:
---bez selekcije---
a-Record Status
b-Record Type
c-Bibliographic Level
d-Hierarchic Level

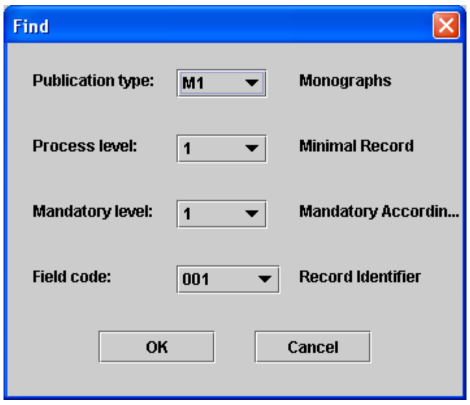
Figure 12. Window Mandatory level subfields

The drop-down list **Mandatory level** contains all entered mandatory level values which belong to the chosen publication type and processing level. When the user chooses the desired mandatory level value which, together with the chosen publication type processing level values has already defined mandatory level subfields, the remaining window components display the corresponding values of the instance whose publication type value is equal to the chosen publication type value, the processing level value is equal to the chosen processing level value, and the mandatory level value is equal to the chosen mandatory level value. Otherwise, the user is automatically transferred to the mode for defining mandatory level subfields.

The drop-down list **Field code** contains all entered field codes. When the user chooses the field code which is already associated to the chosen publication type and processing level and mandatory level values with any of its subfields, the list displaying subfield codes will contain all subfield codes of the chosen field, having subfields already included into the mandatory level subfields marked. Otherwise, the user is automatically transferred to the mode for defining the mandatory level subfields (in the list showing subfield codes there are all subfield codes of the chosen field, but they are not marked).

The list **Subfield code** contains all entered subfield codes of the chosen field, as well as their names. Only the subfields and their names which, together with the chosen publication type values, processing level and mandatory level values and the field code, are already included into mandatory level subfields are marked. While defining mandatory level subfields, when the **OK** button is pressed, it is checked if any of the subfield codes is marked. If neither of them is marked, the corresponding message is shown. Figure 12 shows subfields codes of the *001* field, which are associated to the mandatory level *1* of the processing level *1* and *M1* publication type.

The window for finding the desired instance of the mandatory level subfields concept. This window, which is given in Figure 13, is invoked by pressing the **Preview** button. The user chooses the publication type value. After it is chosen, the user chooses the processing level and mandatory level values and the field code. The desired instance is found after pressing the **OK** button (the field code *001* associated to the mandatory level *1* of the processing level *1* and the *M1* publication type, shown in Figure 13). The action is cancelled by pressing the **Cancel** button.


Find

Publication type:	M1	Monographs
Process level:	1	Minimal Record
Mandatory level:	1	Mandatory Accordin...
Field code:	001	Record Identifier

Figure 13. Window for finding an instance of the mandatory level subfields concept

2.8. Window for Librarian Modification

The window **Librarian**, shown in Figure 14, is used for entering, modification and deleting a librarian.

The single-line editor **Librarian code** receives the entry of/displays the code of a librarian (at most 50 characters). During entering, when the **OK** button is pressed, the check of the type and length of the librarian code value is preformed. If the value is not correct, the corresponding message is shown. It is also checked if the entered code is already present in the database. If so, the corresponding message is shown. After the **Change** button is pressed, it is no longer possible to change the librarian code.

The single-line editor **Internal mark** receives the entry of/displays an internal librarian mark of maximum 10 characters in length. When the **OK** button is pressed during entering and modification, the existence and length of this editor content are checked.

The single-line editor **Secret mark** receives the entry of/displays a secret mark of maximum 6 characters in length. When the **OK** button is pressed during entering and modification, the existence and length of this editor content are checked.

The multi-line editor **Librarian descriptions** receives the entry of/displays the librarian's description at most 240 characters long. Entering a librarian's description is not obligatory.

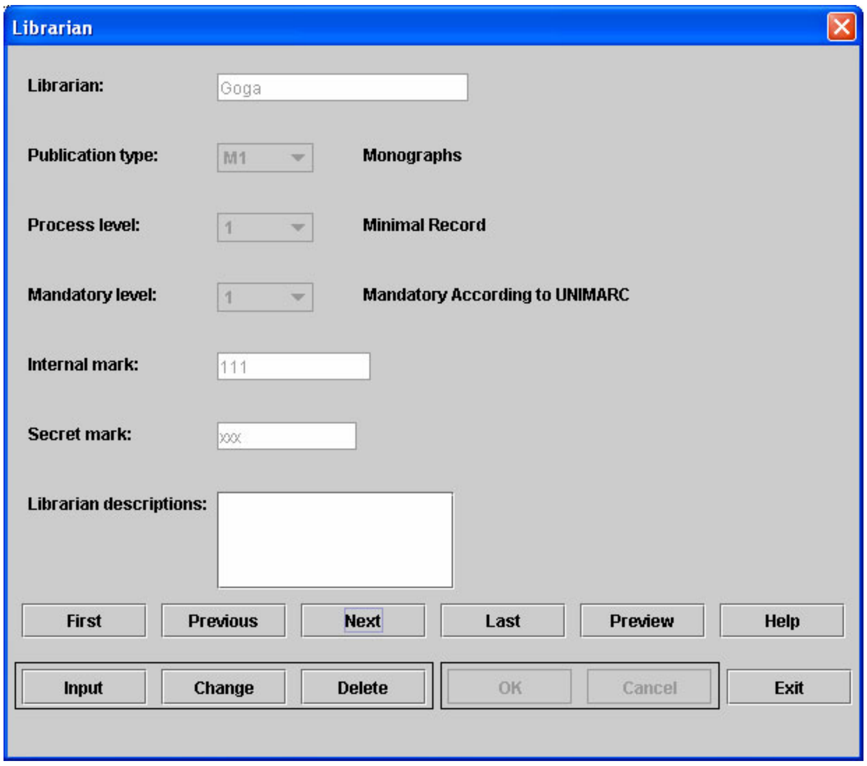
The choice of default values of the publication type, processing and mandatory levels is made by choosing the corresponding values at this dialog for every librarian separately.

Figure 14 shows a librarian with the code *Goga*. The default publication type for this librarian is *M1*, the processing level is *1* and the mandatory level is also *1*. The internal mark is *111*, and the secret code is *xxx*.

Window for finding the desired instance of the librarian concept. This window, given in Figure 15, is invoked by pressing the **Preview** button. The user chooses the desired librarian code. By pressing the **OK** button the desired instance is found (in Figure 15, the librarian code *Goga*). The action is cancelled by pressing the **Cancel** button.

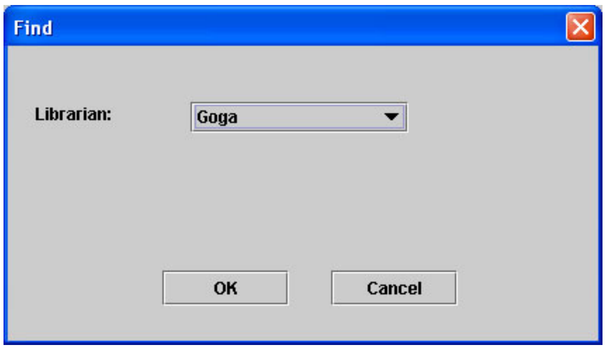
2.9. Window for Librarian Subfields Modification

The window **Librarian subfields**, which is shown in Figure 16, is used for entering, modification and deleting librarian subfields.


Librarian

Librarian:	Goga	
Publication type:	M1	Monographs
Process level:	1	Minimal Record
Mandatory level:	1	Mandatory According to UNIMARC
Internal mark:	111	
Secret mark:	xxx	
Librarian descriptions:		

Figure 14. Window **Librarian**

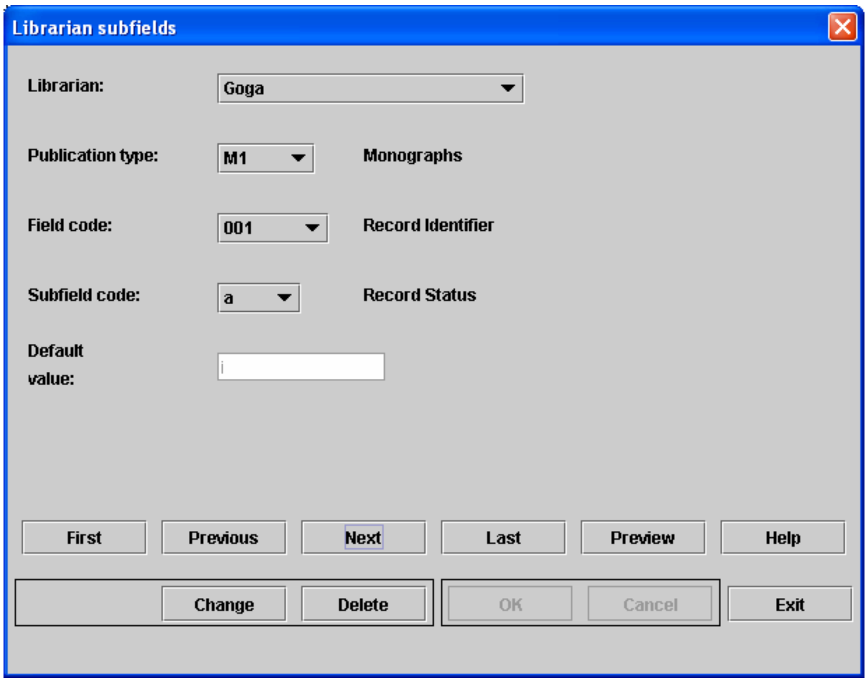

Find

Librarian: Goga

Figure 15. Window for finding an instance of the librarian concept

The drop-down list **Librarian code** contains all entered librarian codes. When the user chooses a librarian code whose librarian subfields are already defined, the remaining window components display the corresponding values of the instance whose librarian code value is equal to the chosen one. Otherwise, the user is automatically transferred to the mode for entering a new instance of librarian subfields.

The drop-down list **Publication type** contains all publication type values which have already defined publication type subfields. When the user chooses the publication type value which has, together with the chosen librarian code, the librarian subfields already defined, the remaining window components display the corresponding values of the instance whose librarian code is equal to the chosen librarian code, and the publication type value is equal to the chosen publication type value. Otherwise, the user is automatically transferred to the mode for entering a new instance of librarian subfields.


Librarian subfields

Librarian: Goga

Publication type: M1 Monographs

Field code: 001 Record Identifier

Subfield code: a Record Status

Default value: i

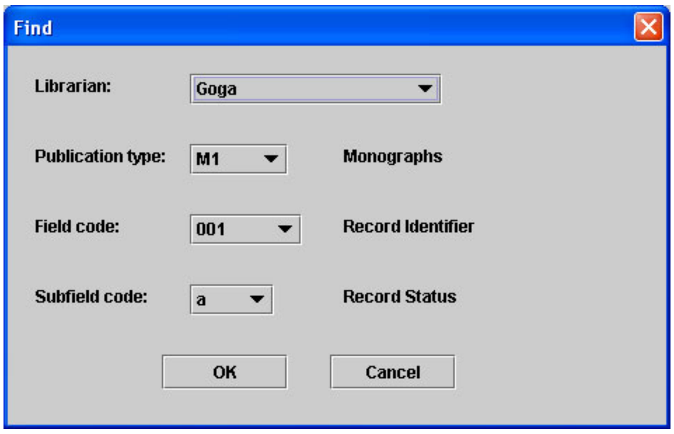
Figure 16. Window **Librarian subfields**

The drop-down list **Field code** contains all entered field codes. When the user chooses a field code which is already associated to the chosen librarian code and publication type with any of its subfields, the remaining window components display the corresponding values of the instance whose librarian code is equal to the chosen librarian code, the publication type value is equal to the chosen publication type value and the field code is equal to the chosen one. Otherwise, the user is automatically transferred to the mode for entering a new instance of librarian subfields.

The list **Subfield code** contains all entered subfield codes of the chosen field. When the user chooses the subfield code which is already associated to the chosen librarian code and the publication type value with the chosen subfield code, the editor where the default subfield value is shows the corresponding value of the instance whose librarian code is equal to the chosen librarian code, the publication type value is equal to the chosen publication type value, the field code is equal to the chosen field code, and the subfield code is equal to the chosen subfield code. Otherwise, the user is automatically transferred to the mode for entering a new instance of the librarian subfield.

The single-line editor **Default value** receives the entry of/displays the default subfield value. If the length of the subfields is defined, during entering and modifying, the check of the length of the entered default value is performed when the **OK** button is pressed.

Figure 16 shows the default value *i* of the subfield *a* of the filed *001*, which are associated to the librarian *Goga* for the publication type value *M1*.


Find

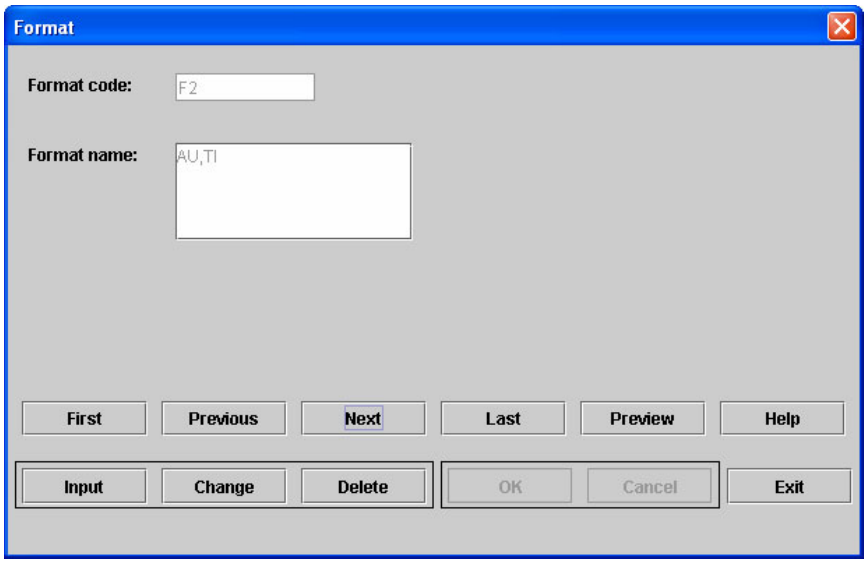
Librarian:	Goga	
Publication type:	M1	Monographs
Field code:	001	Record Identifier
Subfield code:	a	Record Status

Figure 17. Window for finding an instance of the librarian subfields concept

Window for finding the desired instance of the librarian subfield concept. This window, which is shown in Figure 17, is invoked by pressing the **Preview** button. The user chooses the desired librarian code in the component containing librarian codes. After having chosen the librarian code, the user chooses the publication type value, field code and subfield code. The desired instance is found by pressing the **OK** button (in Figure 17 subfield code *a* of the field *001* for the librarian code *Goga* and the publication type *M1*). The action is cancelled by pressing the **Cancel** button.

2.10. Dialog for Format Modification

The window **Format**, shown in Figure 18, is used for entering, modification and deleting formats.



Format code: F2

Format name: AU,TI

Figure 18. Window **Format**

The single-line editor **Format code** receives the entry of/displays the format code (at most 6 characters). During entering, when the **OK** button is pressed, the check of the type and length of the format code is preformed. If the value is not correct, the corresponding message is shown. It is also checked if the entered format code is already present in the database. If so, the corresponding message is shown. After the **Change** button is pressed, it is no longer possible to change the format code.

The multi-line editor **Format name** receives the entry of/displays the format name at most 160 characters long. When the **OK** button is pressed during entering and modification, the check of the existence and length of the content of this component is performed. Figure 18 presents the format code *F2*, whose title is *AU, TI.*

Window for finding the desired instance of the format concept. This window, shown in Figure 19, is invoked by pressing the **Preview** button. The user chooses the format code. By pressing the **OK** button the desired instance is found (the format code *F2*, in Figure 19). The action is cancelled by pressing the **Cancel** button.

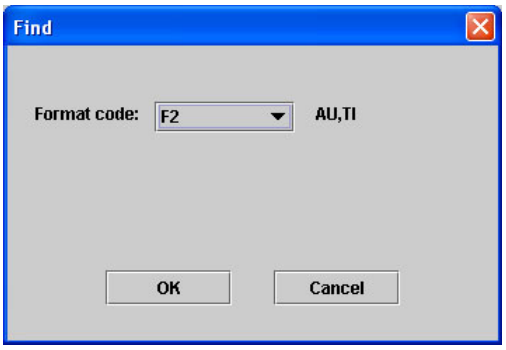


Figure 19. Window for finding an instance of the format concept

2.11. Window for Search Prefixes Modification

The window **Search prefixes**, given in Figure 20, is used for viewing search prefixes. Regarding the actions related to modification, the only one at his/her disposal is prefix name modification, since adding a new prefix affects the text server implementation. In the single-line editor **Prefix code** the prefix code is shown (2 characters).

In the multi-line editor **Prefix name** the prefix name of at most 160 characters is displayed. When the **OK** button is pressed during modification, the existence and length of this editor content are checked.

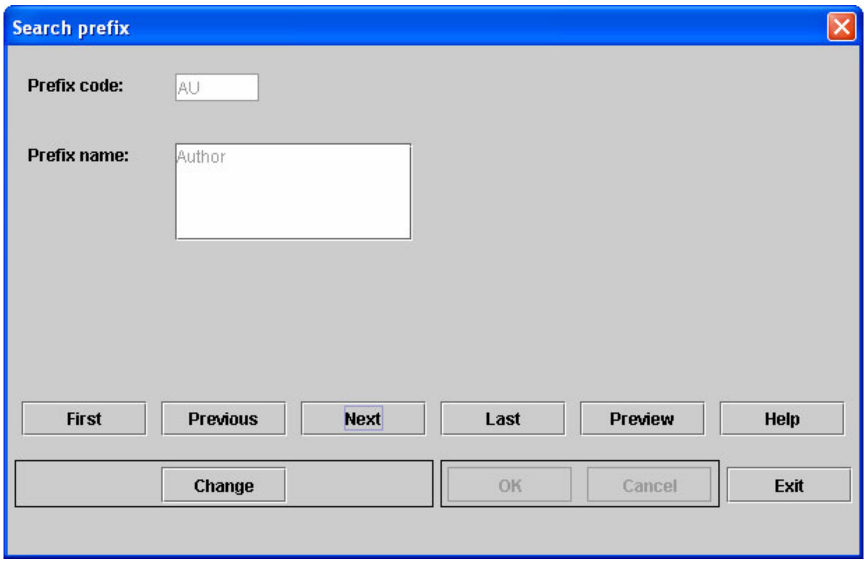


Figure 20. Window **Search prefixes**

Figure 20 shows the *AU* prefix code, whose name is *Author.*

Window for finding the desired instance of the search prefixes concept. This window, shown in Figure 21, is invoked by pressing the **Preview** button. The user chooses the prefix code. The desired instance is found after pressing the **OK** button (prefix code *AU* in Figure 21). The action is cancelled by pressing the **Cancel** button.

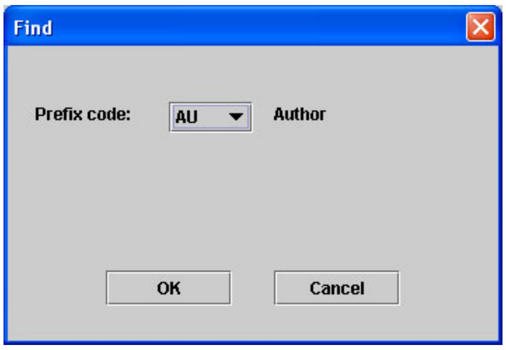


Figure 21. Window for finding an instance of the search prefixes concept

2.12. Window for Format Prefixes Modification

The window **Format prefixes**, given in Figure 22, is used for defining format prefixes.

The drop-down list **Format code** contains all entered format codes. When the user chooses the format code which has already defined search prefixes, in the list which shows prefix codes are all prefix codes, and the prefixes associated to the chosen format are marked. Otherwise, the user is automatically transferred to the mode for defining format prefixes.

The list **Prefix code** contains all entered prefix codes, as well as their names. Only the prefixes and their names which are associated to the chosen format code are marked. During defining a format prefix, when the **OK** button is pressed, it is checked whether at least one prefix code is marked. If none of the prefix codes are marked, the corresponding message is shown. Figure 22 shows prefix codes which are associated to the *F2* format.

Window for finding the desired instance of the format prefixes concept. This window, shown in Figure 23, is invoked by pressing the **Preview** button. The user chooses the desired format code. After pressing the **OK** button, the desired instance is found (the *F2* format code, in Figure 23). The action is cancelled by pressing the **Cancel** button.

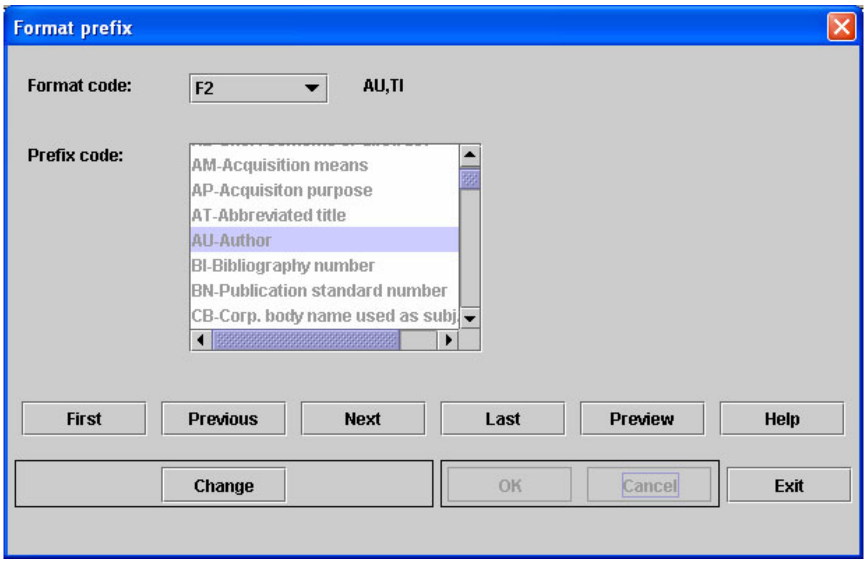


Figure 22. Window **Format prefixes**

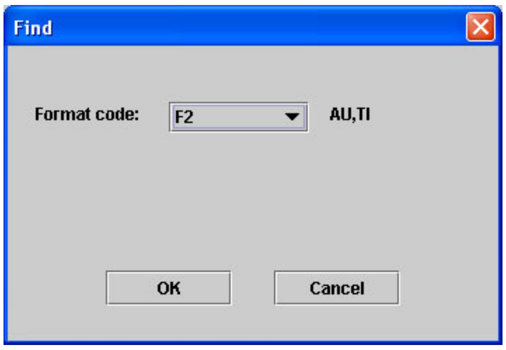


Figure 23. Window for finding an instance of the format prefixes concept

2.13. Window for Search Prefix Subfields Modification

The window **Search prefix subfields**, which is shown in Figure 24, is used for viewing search prefix subfields only, since adding new subfields for a search prefix affects the text server implementation.

The drop-down list **Prefix code** contains all entered search prefixes.

The drop-down list **Field code** contains all entered field codes.

The list **Subfield code** contains all entered subfield codes which belong to the chosen field, as well as their names. The subfields and their names, already associated to the chosen prefix code together with the chosen field code, are marked.

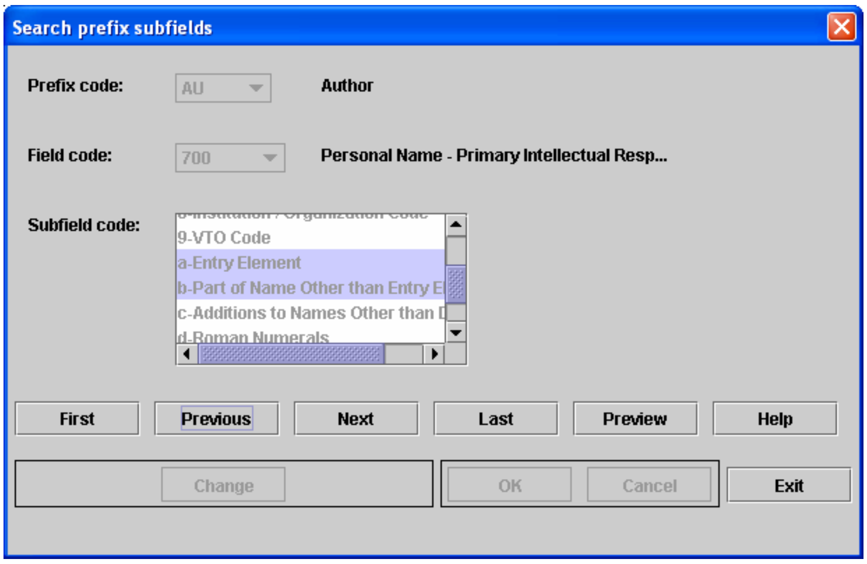


Figure 24. Window **Search prefix subfields**

Figure 24 shows the codes of the subfields of the field *700*, which are associated to the *AU* prefix.

Window for finding the desired instance of the search prefix subfields. This window, presented in Figure 25, is invoked by pressing the **Preview** button. The user chooses the prefix code. After having done it, he/she chooses the field code. After pressing the **OK** button, the desired instance is found (as shown in the example of Figure 25, the field code *700* associated to the *AU* prefix). The action is cancelled by pressing the **Cancel** button.

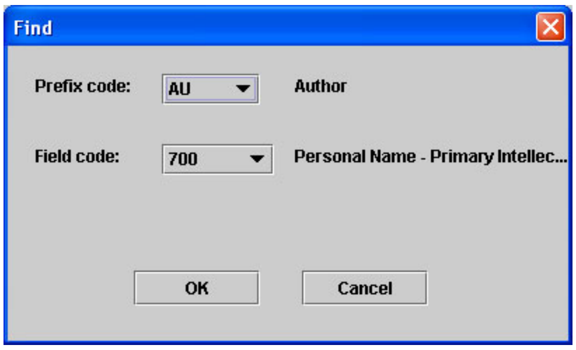


Figure 25. Window for finding an instance of the search prefix subfields concept

2.14. Window for Processing Type Modification

The window **Processing type**, shown in Figure 26, is used for defining the processing type.

The drop-down list **Librarian code** contains all entered librarian codes. Every librarian has at least one processing type representing the default processing type for him/her and which is defined when entering a new librarian using the dialog shown in Figure 14. A new processing type for a chosen librarian is set choosing the publication type value, processing level and mandatory level values other than the ones which are part of the default processing type.

The drop-down list **Publication type** contains all values of the publication type defined for at least one mandatory level. When the user chooses the publication type value included in the processing type for the chosen librarian, the remaining window components display the corresponding values of the instance whose librarian code is equal to the chosen librarian code, and the publication type value is equal to the chosen one. Otherwise, the user is automatically transferred to the mode for defining the processing type.

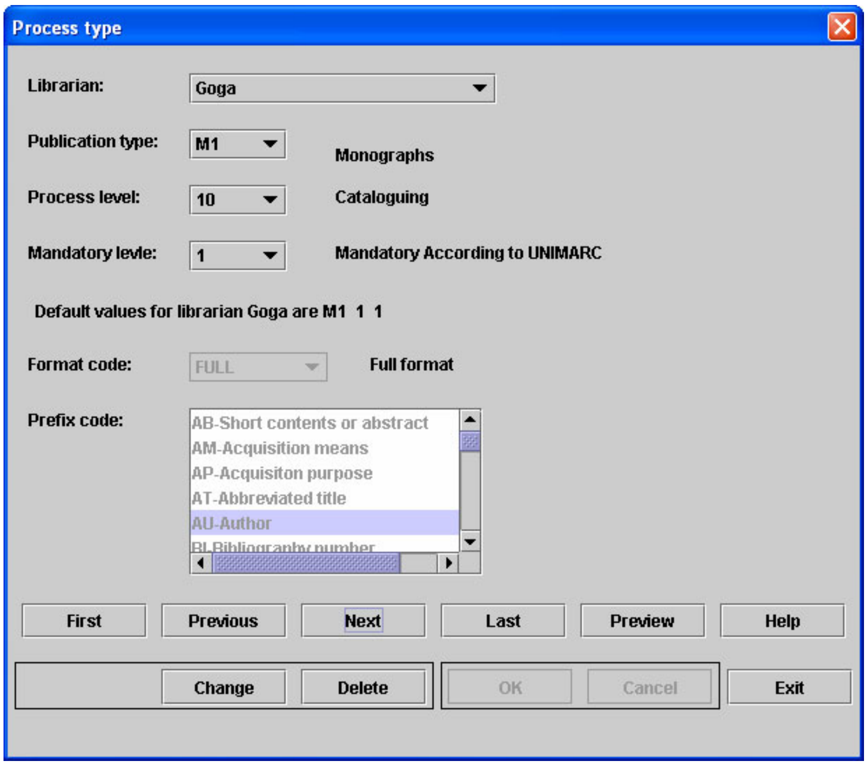
The drop-down list **Processing level** contains all entered values of the processing level for the given publication type for which at least one mandatory level is defined. When the user chooses the processing level value which is, together with the chosen publication type value, included in the processing type of the chosen librarian, the remaining window components display the corresponding values of the instance whose librarian code is equal to the chosen librarian code, value of the publication type is equal to the chosen publication type value, and the processing level value is also equal to the chosen one. Otherwise, the user is automatically transferred to the mode for defining the processing type.

The drop-down list **Mandatory level** contains all entered values of the mandatory level for the given publication type and processing level. When the user chooses the mandatory level value which is, together with the chosen publication type and processing level values, included in the processing type of the chosen librarian, the remaining window components display the corresponding values of the instance whose librarian code is equal to the chosen librarian code, value of the publication type is equal to the chosen publication type value, the processing level value is also equal to the chosen one, and the mandatory level is equal to the chosen mandatory level. Otherwise, the user is automatically transferred to the mode for defining the processing type.

For each chosen librarian the dialog shows the publication type value, processing and mandatory level values which represent the default value of the processing type of that librarian.

The drop-down list **Format code** contains all entered format codes. The user can choose a format code when defining a new processing type for a librarian. Also, when modifying the current processing type for a librarian, the user can perform the format change by choosing any of the entered format codes.

The list **Prefix code** contains all entered prefix codes, as well as their titles. The prefix codes chosen to be presented to the librarian on the mask for search in the application for entering, are marked. When the **OK** button is pressed during entering or modification, the check is performed to check if exactly five prefixes are marked. If that is not the case, the corresponding message is shown.


Process type

Librarian: Goga

Publication type: M1 Monographs

Process level: 10 Cataloguing

Mandatory levle: 1 Mandatory According to UNIMARC

Default values for librarian Goga are M1 1 1

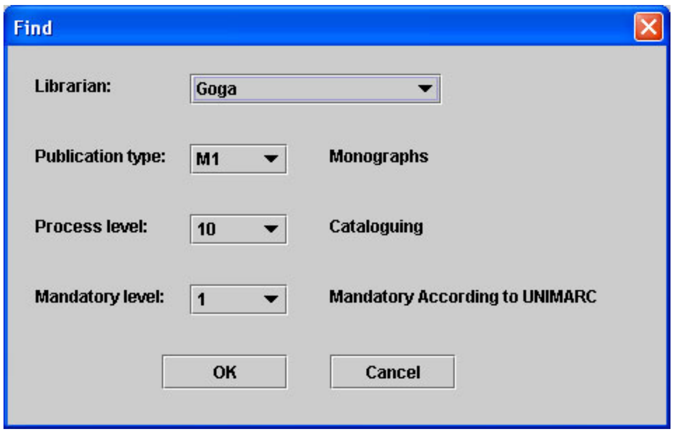
Format code: FULL Full format

Prefix code:
AB-Short contents or abstract
AM-Acquisition means
AP-Acquisiton purpose
AT-Abbreviated title
AU-Author
BI-Bibliography number

Figure 26. Window **Processing type**

Figure 26 shows the processing type for the librarian with the *Goga* code. The publication type value is *M1*, processing and mandatory level values are *10* and *1*, respectively. The format code is *FULL*, and the prefix code *AU* is one of the five prefix codes which must be chosen when defining the processing type for the chosen librarian.

Window for finding the desired instance of the processing type concept. This window, shown in Figure 27, is invoked by pressing the **Preview** button. The user chooses the desired librarian in the drop-down list with librarian codes. After having chosen the librarian, the user chooses the publication type value, processing and mandatory level values. By pressing the **OK** button the desired instance is found (in Figure 27, the librarian *Goga*, the publication type *M1*, processing level *10* and the mandatory level *1*). The action is cancelled by pressing the **Cancel** button.


Find

Librarian: Goga

Publication type: M1 Monographs

Process level: 10 Cataloguing

Mandatory level: 1 Mandatory According to UNIMARC

Figure 27. Window for finding an instance of the processing type concept

3. Conclusion

The application for modifying the Librarian working environment is an integral part of the BISIS ver. 3.0 software system. It enables modification of the data used for the purpose of adapting the application for bibliographical material processing to each librarian's particular requirements, as well as document types being processed by him/her. Furthermore, particular requirements of publication processing are supported, as well as hierarchical publication processing, all according to the needs of a concrete library.

References

[1] *Creating and searching databases in the system of scientific and technological information*, Editor Branislav Lazarević, Ministry of Science and Technology of the Republic of Serbia, Belgrade 1996. (in Serbian)

[2] Pupovac, B., Modeling and implementing the library information system commands, M. S. Thesis, University of Novi Sad, 2000.

[3] Pupovac, B., Librarian work environment, INFOteka, 2 (2001) 1-2, Beograd, pp. 105-109

[4] Eckel, B., Thinking in Java, 2nd edition, 2000., http://www.EckelObjects.com

[5] Java^{TM} 2 Platform, Standard Edition, v 1.3.1 API Specification, http://java.sun.com/j2se/1.3/docs/api/

[6] Pupovac, B., *Maintenance of the YUMARC Format*, Intl. Conf. on Distributed Library Information Systems, Ohrid, 2004.

User Search in the Library Information System BISIS

Branko Milosavljević
Faculty of Technical Sciences, University of Novi Sad
mbranko@uns.ns.ac.yu

Abstract. The user search subsystem of the library information system BISIS provides access to the bibliographic catalogue via the web to library members and visitors. This paper presents the functionality and outlines the software architecture of the web site for user search. Searching functionality and features are described in detail and illustrated with multiple examples on real bibliographic data. The software architecture is presented in brief and the implementation platform is specified.

1. Introduction

User search is a software module of the Library Information System BISIS [1] providing search and retrieval of bibliographic records via the web. Traditionally this function of library information systems has been named OPAC (online public access catalogue). The main users of this web site are library visitors or members. The home page of BISIS web site is presented in Fig. 1. It provides a choice of the language used for displaying textual messages during the interaction with the site for the current user session. By clicking *Srpski* and *English* the user is able to choose between Serbian and English language, respectively. The home page also offers several options for initiating the search on bibliographic records. It is possible to express simple queries (on a single search criteria, like author or title) and more complex queries comprising multiple search criteria.

The main menu, consisting of links contained in the home page, has several options for searching the bibliographic database handled by BISIS. The options are as follows:

- *by author* – searching by author's last and/or first name,
- *by title* – searching by words contained in the title of the work,
- *by author and title* – searching by author's name and the title at the same time,
- *by keywords* – searching by keywords comprising several UNIMARC subfields, and

- *custom query* – searching by at most five different prefixes combined with logical (Boolean) operators.

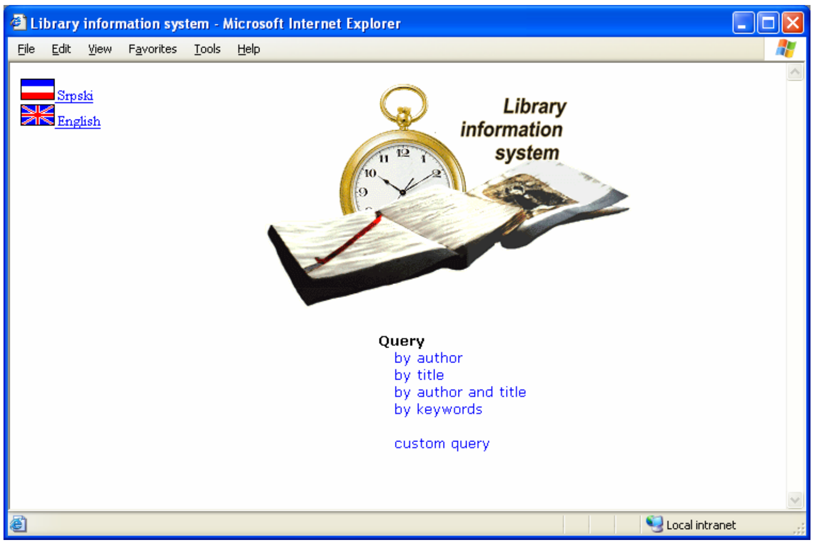
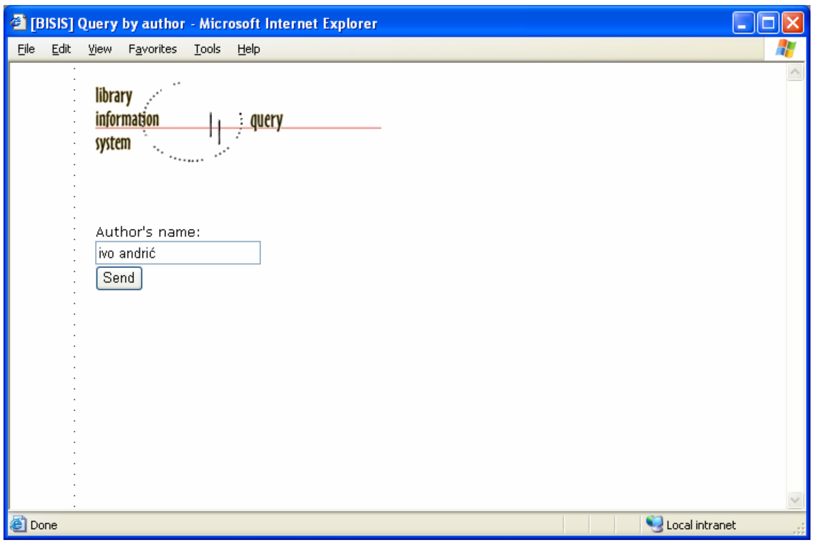

Library information system - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Srpski
English
Library information system
Query
by author
by title
by author and title
by keywords
custom query
Local intranet

Figure 1. The web site home page

2. Searching by a single prefix

Following the link *by author* a page presented in Fig. 2 is displayed.


[BISIS] Query by author - Microsoft Internet Explorer
File Edit View Favorites Tools Help
library information system
query
Send
Done
Local intranet

Author's name: ivo andrić

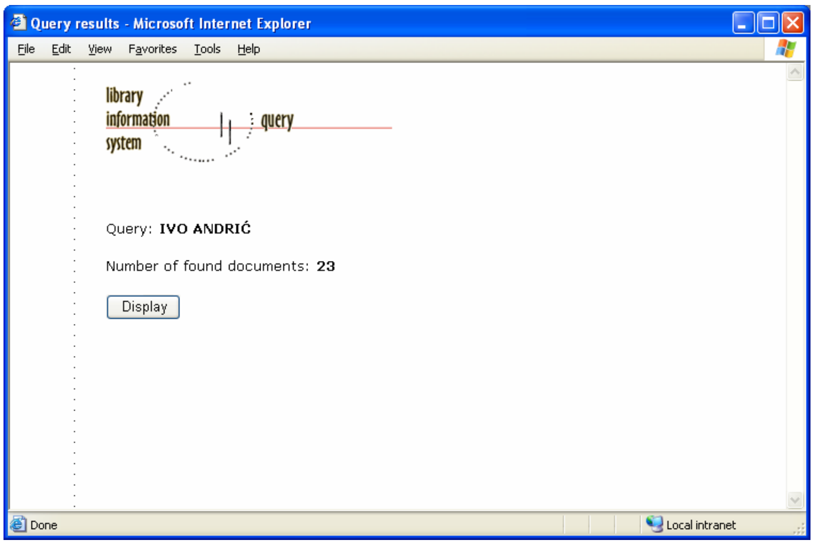
Figure 2. A page for searching by author

This page contains a text field *Author's name* that is to be filled with the desired name. The author's first and last name can be entered in any order. Besides, searching by author, like any other means of searching the bibliographic database on the BISIS web site, is characterised by the following features:

- the system is case-insensitive for searching textual data,
- punctuation marks are ignored,
- searching is equivalently performed using cyrillic and latin alphabets for text content written in Serbian,
- accented characters of the Serbian latin alphabet (šćčžđ) can be replaced by non-accented base characters (scczdj), and
- available wildcard characters are '?' (replaces a single character) and '*' (replaces a sequence of characters).

For example, queries *Ivo Andrić*, *ivo andrić*, *andrić ivo* and *andrić, ivo* are equivalent. The result of the query *ivo andric* (a non-accented version) will comprise, besides the records with the given content, all hits found in previous queries.

The actual process of searching bibliographic database is initiated by clicking the *Send* button upon entering the query. A page with the number of records matching the query is displayed as a result. An example page is presented in Fig. 3.


Query results - Microsoft Internet Explorer
File Edit View Favorites Tools Help
library information system
query
Display
Done
Local intranet

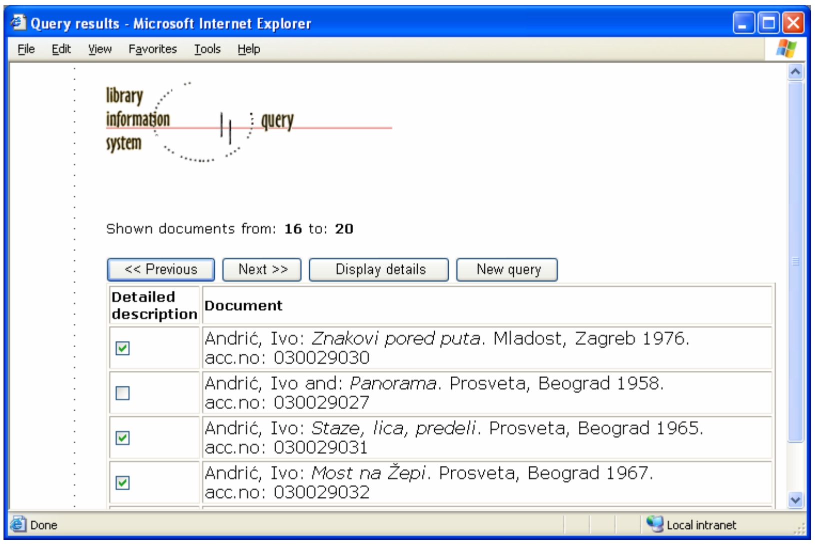
Query: IVO ANDRIĆ

Number of found documents: 23

Figure 3. The number of records matching the query

The user can now choose to browse through the result set by clicking the *Display* button. Then the user is directed to a page with matching records presented in a brief format, as presented in Fig. 4. The result list is divided into segments comprising at most five hits. The segments can be bidirectionally traversed from the beginning to the end. Traversal is invoked by clicking the << *Previous* and *Next* >> buttons for traversing towards the beginning and the end, respectively. By clicking the *New query* button the user navigates to the home page in order to formulate a new query.

Bibliographic records matching the query presented in the brief display mode are listed in a table that contains a check box *Detailed description* associated with each item. The user can check the *Detailed description* box for the given item thus marking it for the detailed display. This selection may span several brief list fragments. After all the records intended for detailed display are marked, the user can switch to the detailed display by clicking the *Display details* button.


Query results - Microsoft Internet Explorer
File Edit View Favorites Tools Help
library information system query
<< Previous
Next >>
Display details
New query
Done
Local intranet

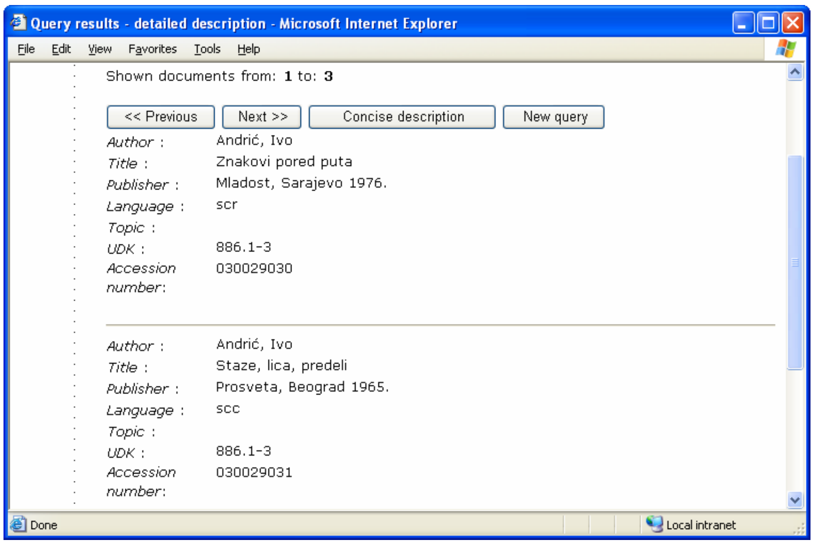
Shown documents from: **16** to: **20**

Detailed description	Document
✓	Andrić, Ivo: *Znakovi pored puta*. Mladost, Zagreb 1976. acc.no: 030029030
☐	Andrić, Ivo and: *Panorama*. Prosveta, Beograd 1958. acc.no: 030029027
✓	Andrić, Ivo: *Staze, lica, predeli*. Prosveta, Beograd 1965. acc.no: 030029031
✓	Andrić, Ivo: *Most na Žepi*. Prosveta, Beograd 1967. acc.no: 030029032

Figure 4. A brief display of records matching the query

The page containing a detailed display of marked records is presented in Fig. 5. Detailed display presents more data from the actual record to the user. Similarly to the brief display mode, if the marked record list contains more than five items, the list is fragmented into 5-piece segments. Bidirectional traversal of the detailed display list is provided by << *Previous* and *Next* >> buttons. Clicking the *Concise description* button navigates the user back to the brief display. Clicking the *New query* button directs the user to the home page.

Starting from the home page, the user can follow the *by author and title* link and initiate the search by the author and the title at the same time. Figure 6 presents the page with an example of such a query where the text field *Author's name* contains the text *andrić, ivo* while the *Title* field contains the text *на дрини ћуприја* (the cyrillic equivalent of *na drini ćuprija*). Since the retrieval using cyrillic and latin alphabets is equivalent for content written in Serbian, the result will comprise all bibliographic records containing the given text in any of the two alphabets. In this example, the page in Fig. 7 presents the number of records matching the query, while the page in Fig. 8 presents a list of hits in brief display mode. Presentation of the data contained a bibliographic record uses the alphabet actually used in the record itself.



Query results - detailed description - Microsoft Internet Explorer
File Edit View Favorites Tools Help
<< Previous
Next >>
Concise description
New query
Done
Local intranet

Shown documents from: **1** to: **3**

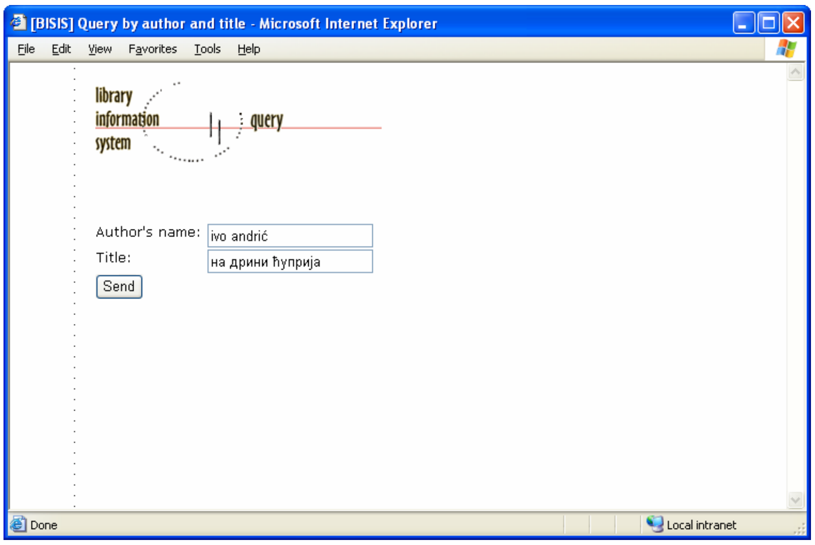
Author :	Andrić, Ivo
Title :	Znakovi pored puta
Publisher :	Mladost, Sarajevo 1976.
Language :	scr
Topic :	
UDK :	886.1-3
Accession number:	030029030

Author :	Andrić, Ivo
Title :	Staze, lica, predeli
Publisher :	Prosveta, Beograd 1965.
Language :	scc
Topic :	
UDK :	886.1-3
Accession number:	030029031

Figure 5. Detailed display of selected records

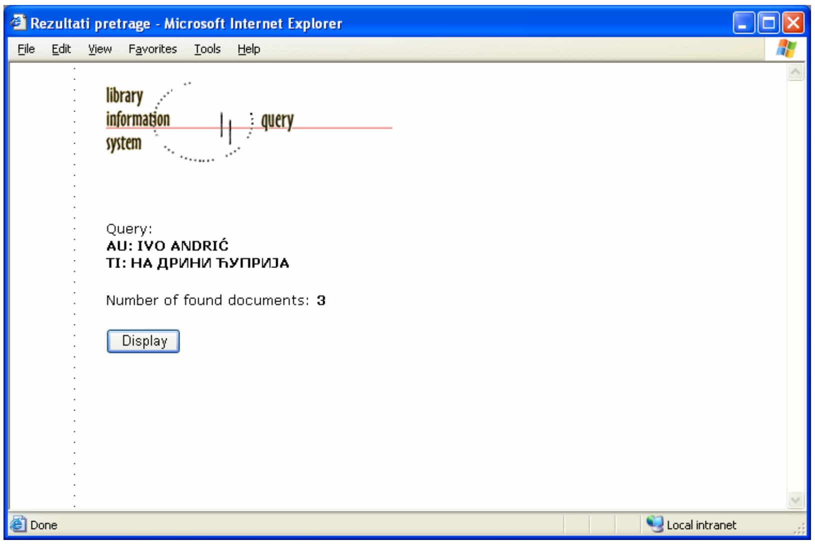
3. Custom Queries

The link *custom query* on the home page invokes the page for formulating queries that combine multiple prefixes presented in Fig. 9.


[BISIS] Query by author and title - Microsoft Internet Explorer
File Edit View Favorites Tools Help
library information system query
Send
Done
Local intranet

Author's name: ivo andrić
Title: на дрини ћуприја

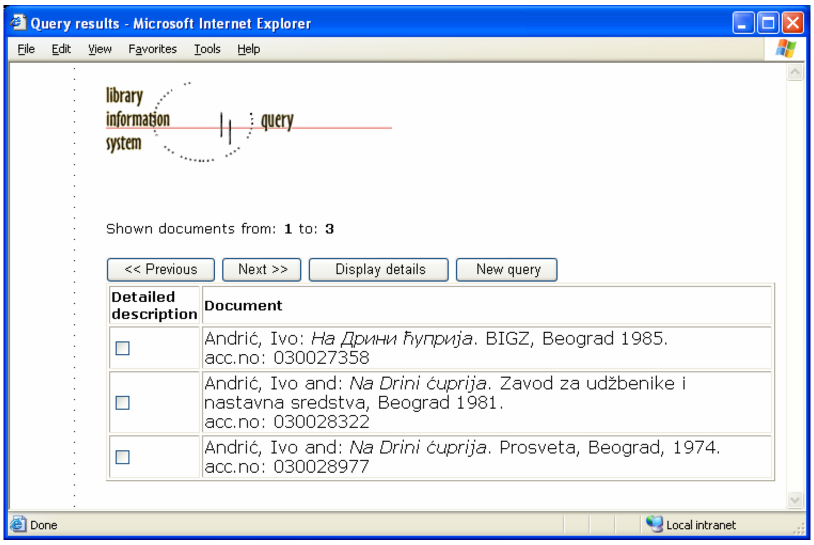
Figure 6. Searching by the author and the title simultaneously


Rezultati pretrage - Microsoft Internet Explorer
File Edit View Favorites Tools Help
library information system query
Display
Done
Local intranet

Query:
AU: IVO ANDRIĆ
TI: НА ДРИНИ ЋУПРИЈА

Number of found documents: 3

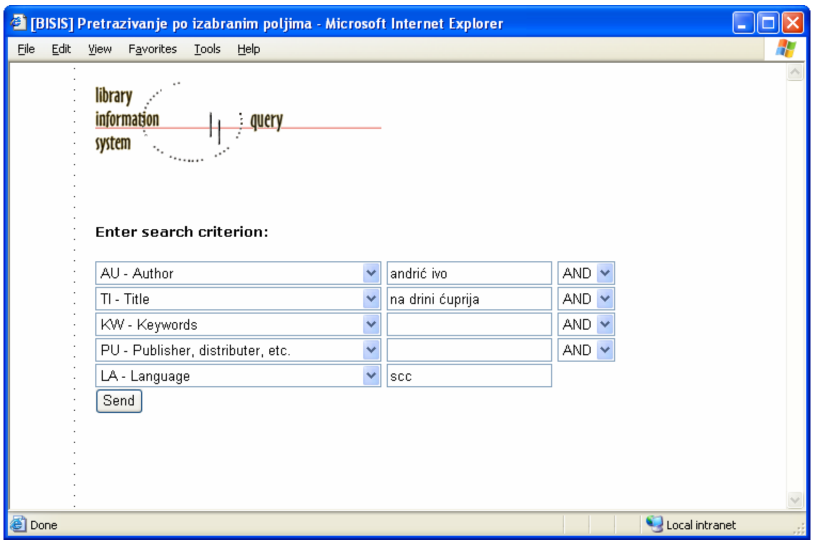
Figure 7. The result of a search using different alphabets


Query results - Microsoft Internet Explorer

Shown documents from: **1** to: **3**

Detailed description	Document
☐	Andrić, Ivo: *На Дрини ћуприја*. BIGZ, Beograd 1985. acc.no: 030027358
☐	Andrić, Ivo and: *Na Drini ćuprija*. Zavod za udžbenike i nastavna sredstva, Beograd 1981. acc.no: 030028322
☐	Andrić, Ivo and: *Na Drini ćuprija*. Prosveta, Beograd, 1974. acc.no: 030028977

Figure 8. The result set display uses the original alphabet used in the record


[BISIS] Pretrazivanje po izabranim poljima - Microsoft Internet Explorer

Enter search criterion:

AU - Author	andrić ivo	AND
TI - Title	na drini ćuprija	AND
KW - Keywords		AND
PU - Publisher, distributer, etc.		AND
LA - Language	scc	

Figure 9. Forming a custom query

It is possible to combine at most five different prefixes in a custom query. Combo boxes on the left hand side provide the choice of prefixes. The content of the prefix is entered in a text field next to the corresponding prefix (in the same row). Multiple prefixes may be combined by means of logic operators listed in a combo box at the right hand side of each row. Logic operators are: AND (set intersection), OR (set union), and NOT (set difference). If a text field is left blank, the given line is ignored in the query. An example of a custom query, presented in Fig. 9, expresses the following searching requirements:

- the author of a bibliographic item is Ivo Andrić (the first combo box displays the selected prefix *AU – Author*, and the corresponding text field contains the text *andrić ivo*),
- the title of a bibliographic item **is not** “Na Drini ćuprija” (the operator combo box in the previous non-empty line displays the selection *NOT*, the prefix to be searched is selected in the current prefix combo box and it is *TI – Title*, and the corresponding text field contains the text *na drini ćuprija*), and
- the language of the bibliographic item is Serbian, using cyrillic alphabet (the fifth row contains the text *scc*, the coded name for Serbian language, the chosen prefix is *LA – Language*, and the operator combo box in the previous non-empty row displays the selection *AND*).

Let us observe the process of entering this example query. The first part of the query defines searching by author. The prefix representing the document's author is AU and it is chosen in the first combo box, as presented in Fig. 10. The text field in the same row contains the name of the author. In this example, it is Ivo Andrić, as presented in Fig. 11.

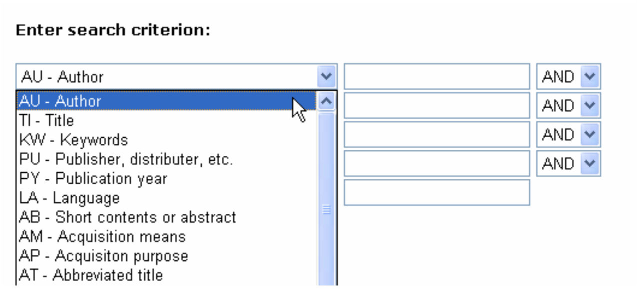


Figure 10. Choosing the prefix AU from the combo box

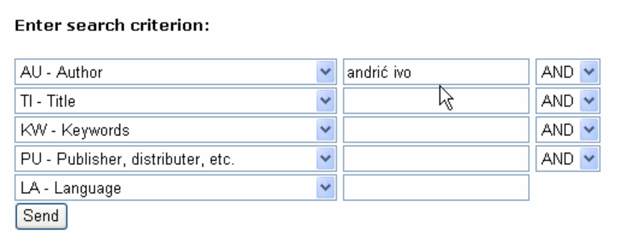
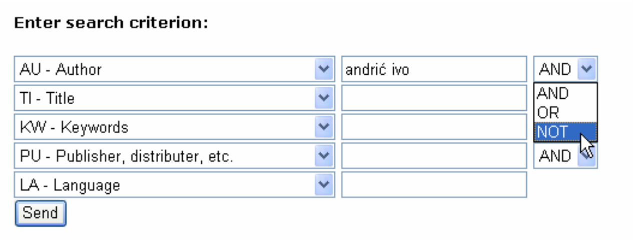


Figure 11. Entering the requested content of the prefix AU

The next part of the example custom query is the statement that the document title **does not** contain the text *Na drini ćuprija*. The logic negation is represented by the logic operator *NOT* connecting the previous part of the query and the current part. This operator is chosen in the row belonging to the previous part of the query (the first row in this example, as presented in Fig. 12). Operator *a NOT b* connects two parts of the query in such way that the result contains records having the first given content (a) and not having the second one (b).

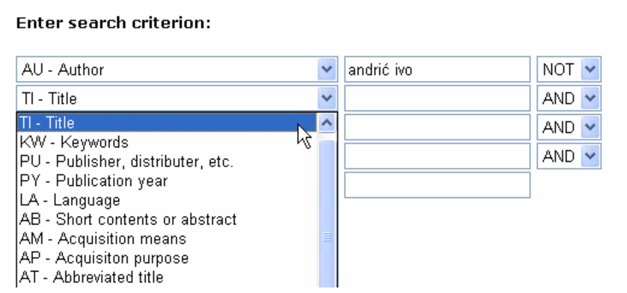


Enter search criterion:

AU - Author	andrić ivo	AND
TI - Title		
KW - Keywords		
PU - Publisher, distributer, etc.		AND
LA - Language		

Figure 12. The choice of the NOT operator

The second part of the query can be entered in any line of the query form. In this example, it is the second row, as presented in Fig. 13.

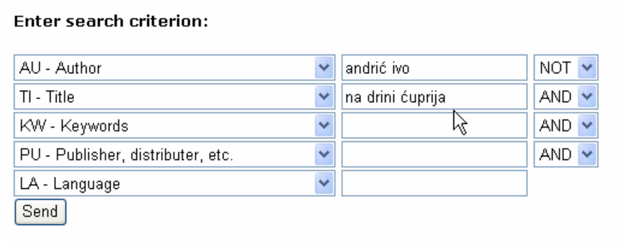


Enter search criterion:

AU - Author	andrić ivo	NOT
TI - Title		AND
		AND
		AND

Figure 13. Choosing the prefix TI

The requested content of the prefix TI is entered in the text field in the same row where the prefix has been chosen. Fig. 14 illustrates this operation.

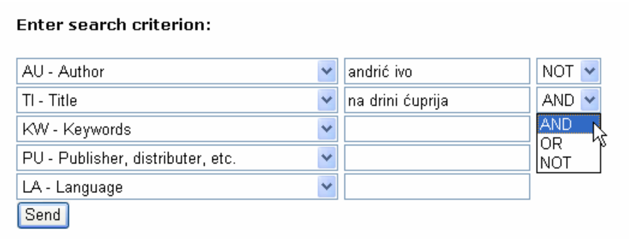


Enter search criterion:

AU - Author	andrić ivo	NOT
TI - Title	na drini ćuprija	AND
KW - Keywords		AND
PU - Publisher, distributer, etc.		AND
LA - Language		

Figure 14. Entering the requested content of the prefix TI

The third part of the example custom query defines the language of the bibliographic item. It is the Serbian language using cyrillic script. This part of the query is connected to the rest of the query by the operator AND. This operator is chosen in the last of the previous non-empty rows – the second row in this example, as illustrated in Fig. 15.

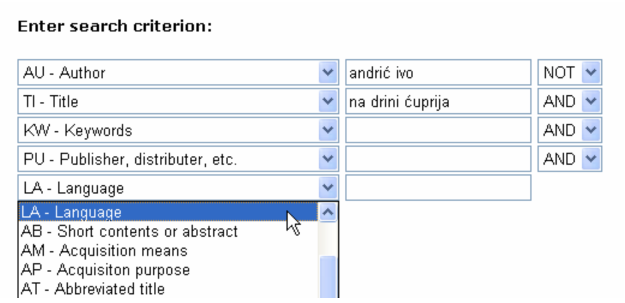


Enter search criterion:

AU - Author	andrić ivo	NOT
TI - Title	na drini ćuprija	AND
KW - Keywords		AND / OR / NOT
PU - Publisher, distributer, etc.		
LA - Language		

Figure 15. The choice of the AND operator

The choice of the prefix LA, representing document language, may be made in any of the rows below the last row used. The example illustrated in Fig. 16. uses the combo box in the fifth row.

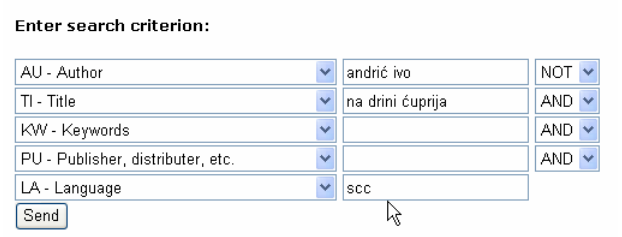


Enter search criterion:

AU - Author	andrić ivo	NOT
TI - Title	na drini ćuprija	AND
KW - Keywords		AND
PU - Publisher, distributer, etc.		AND
LA - Language		

Figure 16. Choosing the prefix LA

Query formulation ends in entering the requested content for the last prefix used. In this example it is the content *scc*, typed into the text field in the last row (next to the corresponding prefix). This situation is illustrated in Fig. 17.



Enter search criterion:

AU - Author	andrić ivo	NOT
TI - Title	na drini ćuprija	AND
KW - Keywords		AND
PU - Publisher, distributer, etc.		AND
LA - Language	scc	

Figure 17. Entering the requested content of the prefix LA

Upon completing the query formulation, the actual searching process is initiated by clicking the *Send* button. The report on the number of records matching the query as well as browsing the result set in brief and detailed display mode is handled the same way as for other types of queries.

Various complex queries may be formulated analogously to the previous example. Retrieval by chosen prefixes facilitates precisely formulated searches for partucular bibliographic records. Table 1 lists examples of complex queries and descriptions of the obtained results.

Query	Description

AU = ivo andrić NOT
TI = na drini ćuprija AND
LA = scc
	Records with the author being *Ivo Andrić*, and the title **not** containing *Na Drini ćuprija*, and the language being Serbian using cyrillic script.

AU = ivo andrić OR
AU = petar džadžić AND
TI = na drini ćuprija
	Records with the author being *Ivo Andrić* or *Petar Džadžić*, and the title containing the text *Na Drini ćuprija*.

AU = ivo andrić AND
TI = na drini ćuprija OR
TI = znakovi pored puta
	Records with the author being Ivo Andrić and the title containing either *Na Drini ćuprija* or *Znakovi pored puta*.

Table 1. Examples of custom queries with their semantics

4. Software architecture

Searching functionality in BISIS is available to multiple types of users. Librarians use special applications, presented in [2], to access and edit the bibliographic database. These applications access the database by means of a text server [3], a software module for indexing and retrieval of bibliographic records. On the other hand, library members and other visitors can access the bibliographic catalogue via the web site for user search using the standard HTTP protocol [4]. The functionality of this web site is presented in the UML use-case diagram in Fig. 18.

The use-case *Query Formulation* represents entering the search request (ie., a query) by the user (a library member or an anonymous visitor). Query formulation is classified by its complexity into two subcases. A *Simple Query* is a facility for creating common, simple, and easy to understand queries that combine at most two prefixes in searching (e.g., author, title, or keywords). A *Complex Query* enables the formulation of sophisticated queries comprising multiple elements combined by Boolean logic operators. This case of query formulation, being a more general one and more complex than the previous, is visually separated from the others in the web site home page.

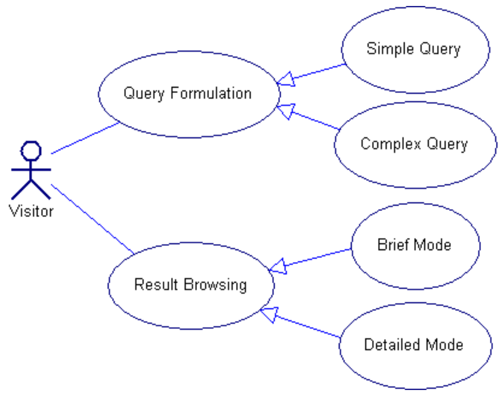


Figure 18. The use-case diagram of the web site for user search

The use-case *Result Browsing* represents a situation when the user has already formed the query, got the report on the number of records found and initiated a result set browsing session. The result set can be browsed in two different display modes – *Brief Mode* and *Detailed Mode*. These two display modes differ in the quantity of data displayed for each record in the result set and the formatting of the displayed data.

Since the web application for user search is used solely for data access rather than data editing, the web server used does not need to implement special functionality for concurrent access and editing data such as connection pooling [5]. Maintaining internal memory cache structures for speeding up access to commonly accessed data is performed by the database server. Hence, the web server or the contained web application for user search does not need to implement this functionality since it is already provided by the database server. On the other hand, information on multiple concurrent user sessions must be maintained in the memory of the web server. These information are used while generating HTML pages sent to particular web site users. The UML deployment diagram in Fig. 19 represents components of the BISIS web application for user search.

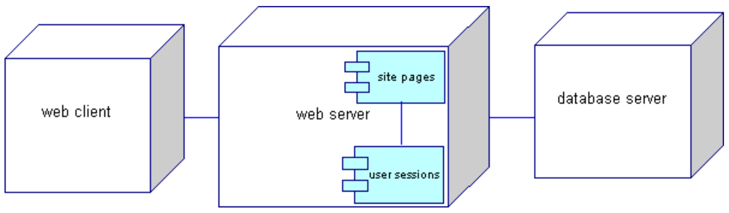


Figure 19. The deployment diagram of the web application for user search

The web site implementation is based on the Java programming language and the set of related technologies for web site construction, namely servlets [6] and JSP [7] for serving dynamic HTML content. The use of the Java platform enables installation of the web server on different hadrware and software platforms. The current implementation uses Jakarta Tomcat [8], an open source, JSP-compliant web server, as the web server of choice.

5. Conclusions

Web based user interface is characterised by its widespread use and availability. It is represents the most suitable technology for public access systems such as user search in a library information system. The user search subsystem of BISIS, being a web site that provides only viewing the data in the bibliographic database rather than modifying it, is suitable to fit in a simple software architecture for web applications. A web application of this type must be capable of handling a large number of concurrent users at an acceptable level of performance.

The web site structure facilitates access to its functions for users with different levels of skill. Multiple forms for expressing queries, with different complexity, and different result set viewing modes enable users to customize interaction with the system to fit their particular needs and skills.

Further development of the user search subsystem in BISIS is directed towards adding access to other functions of the BISIS system, particularly the functions of document circulation. A notable example is the possibility of making reservations for certain titles via this web site available for registered library members.

References

[1] D. Surla and Z. Konjović, eds. *Modeling and Implementation of the Library Information System BISIS*. Group for Information Technologies, Novi Sad, 2004. (in Serbian)

[2] M. Vidaković, T. Zubić, B. Milosavljević, B. Pupovac, and T. Tošić. *Processing Bibliographic Documents in the Library Information System BISIS*. Proc. of International Conference on Distributed Library Information Systems, Ohrid, FYROM, 2004.

[3] B. Milosavljević. *Text Server for UNIMARC Records*. Master's thesis, Faculty of Technical Sciences, Novi Sad, 1999. (in Serbian). http://diglib.ns.ac.yu/ndltd/docs/set1/ndltd6/textsrv.pdf

[4] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. *Hypertext Transfer Protocol – HTTP/1.1.* IETF RFC 2616. http://www.ietf.org/rfc/rfc2616.txt

[5] E. Roman, S. Ambler, and T. Jewell. *Mastering Enterprise JavaBeans*, 2nd Ed., John Wiley and Sons, New York, 2002.

[6] D. Coward. *Java Servlet Specification Version 2.3*, Sun Microsystems, 2001. http://java.sun.com/j2ee

[7] M. Roth, E. Pelegrí-Llopart. *JavaServer Pages Specification Version 2.0*. Sun Microsystems, 2003. http://java.sun.com/j2ee

[8] *Jakarta Tomcat*. http://jakarta.apache.org/tomcat

Administration of the Library Information System BISIS

Branko Milosavljević, Milan Vidaković
Faculty of Technical Sciences, University of Novi Sad
{mbranko,minja}@uns.ns.ac.yu

Abstract. The Library Information System BISIS is a three-tier internet application that has a number of installation and configuration parameters. Adjusting these parameters, along with the functions of importing and exporting data from a BISIS database and creating backup copies are the main tasks in the BISIS system administration. This paper describes these tasks in detail. A list of all system configuration parameters is given and the processes of importing and exporting data and creating backup copies are presented.

1. Introduction

The Library Information System BISIS [1] is an example of a distributed library information system. A library network consists of nodes, where each node represents an installation of BISIS or a similar system connected to the network via open communication and data protocols. Each node in the network handles a particular library institution and needs to be configured to suit the needs of that institution. This paper presents components of BISIS that need to be configured during the installation of the system. Diagram in Fig. 1. presents an overview of the BISIS software architecture and its main components.

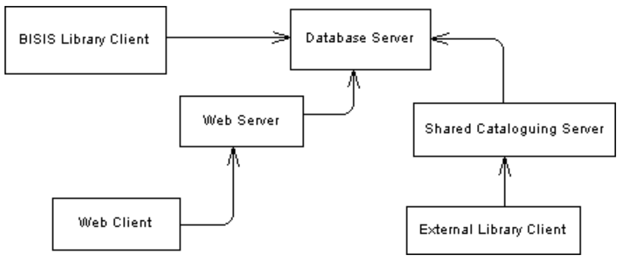


Figure 1. The components of the BISIS system

The installation of BISIS comprises the following tasks:

- installing the database server and creating the BISIS database schema,
- installing the web server,

- installing the shared cataloguing server, and
- installing library clients.

During the exploitation of the system, several administrative tasks may be needed. These include importing and exporting database data, and creating backup copies of the data. The following sections describe those tasks.

2. Configuring the Database Server

Upon installing the database software and creating the physical database, the database schema used by BISIS needs to be created within the database. A special application in the BISIS client application set, *DB Assistant*, provides facilities for creating and dropping the database schema used by BISIS. The main application window is presented in Fig 2. Clicking the *Server...* button in the main application window opens the dialog for adjusting server connection settings. This dialog is presented in Fig. 3. The operation to be performed may be chosen in the *Choose operation* radio button set.

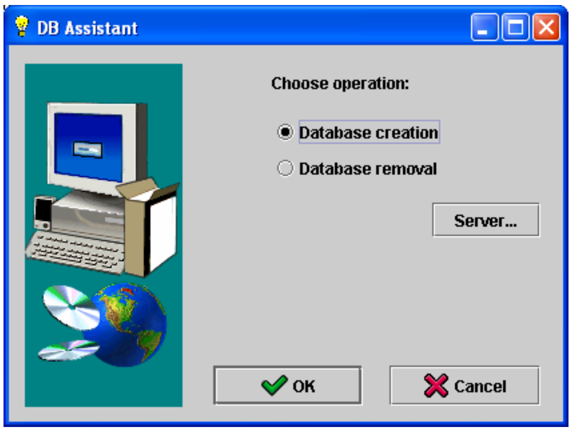
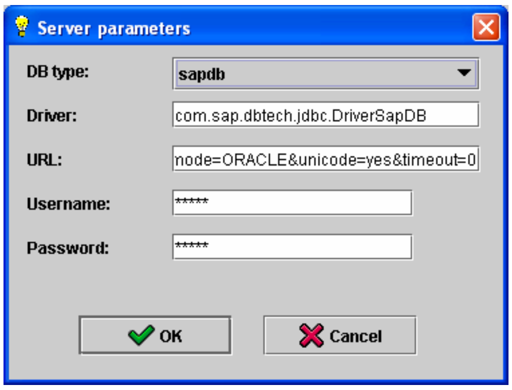


Figure 2. DB Assistant

Once the server connection parameters are set, clicking the *OK* button in the main application window will invoke the chosen operation.

3. Configuring the Web Server

A web server suitable for running a BISIS web site is any web server compliant with servlet [2] and JSP [3] specifications. The use of Jakarta Tomcat [4], a reference implementation for servlets and JSP, is recommended. Installing the BISIS web application is performed by copying the *bisis.war* file included in the BISIS distribution in the web application tree of the web server. Registration of the web application and defining the URL page context depends on the web server used. The web server will unpack the bisis.war archive after the installation. A file named *Settings.properties* contained in the unpacked web application directory needs to be edited. This file follows the standard Java resource bundle syntax. The parameters in the file and their meaning is listed in Table 1.



Server parameters

DB type: sapdb

Driver: com.sap.dbtech.jdbc.DriverSapDB

URL: node=ORACLE&unicode=yes&timeout=0

Username: *****

Password: *****

Figure 3. Server connection parameters

Parameter	Description
type	The type of the database. For example, Oracle database is denoted by *oracle*, while SAP DB is denoted by *sapdb*.
driver	The name of the Java class representing the appropriate JDBC database driver. For example, Oracle database uses a driver named *oracle.jdbc.driver.OracleDriver*, while SAP DB database uses a driver named *com.sap.dbtech.jdbc.DriverSapDB*.
url	The JDBC address of the database used. The syntax of this URL depends on the type of the database used.
username	The username used for accessing the database.
password	The password used for accessing the database.
locale	Initial locale for new user sessions. Valid values are two-letter language abbreviations listed in the *ISO 639 Language Codes* [5].
maxHits	The maximum number of hits presented in a single page. The default is 5.
entryFields	The number of available prefix lists in the page for custom query formulation. The default is 5.
SMTP_address	Address of an SMTP server used for sending email messages.
SMTP_port	The TCP port of an SMTP server used for sending email

	messages.
SMTP_reply_to	The email address users can reply to.
defaultAddress	The email address used for denoting the sender of email messages sent from the BISIS web site.

Table 1. The BISIS web application parameters

4. Configuring the Shared Cataloguing Server

The Shared Cataloguing Server is a standalone Java server application that is contained in the file *sharedsrv.jar* in the BISIS software distribution. Its configuration is stored in the [sharedsrv] section of the *config.ini* file contained in the same directory as the *sharedsrv.jar* file. The [sharedsrv] section contains several parameters listed in Table 2 along with their descriptions.

Parameter	**Description**
type	The type of the database. For example, Oracle database is denoted by *oracle*, while SAP DB is denoted by *sapdb*.
driver	The name of the Java class representing the appropriate JDBC database driver. For example, Oracle database uses a driver named *oracle.jdbc.driver.OracleDriver*, while SAP DB database uses a driver named *com.sap.dbtech.jdbc.DriverSapDB*.
url	The JDBC address of the database used. The syntax of this URL depends on the type of the database used.
username	The username used for accessing the database.
password	The password used for accessing the database.
biblusername	The librarian username used for accessing the records.
biblpassword	The librarian password used for accessing the records.
validatorClass	The name of the Java class implementing record validation.
preconnectCount	The number of initially open connections to the database.
maxIdleTextServers	The maximum number of idle connections to the database.
maxTextServers	The maximum number of open connections to the database.
timeToCheckDead TextServers	Time period between database connection checks.
limit	The number of failed database connection checks after which a connection is labeled as inactive.
logTreshold	The maximum number of log items.

Table 2. The Shared Cataloguing Server parameters

5. Configuring Client Applications

The BISIS client application set is installed on librarian workstations. Client appplications are contained in a set of *jar* files in the installation directory. Besides the *jar* files, there is a text file named *config.ini* holding configuration parameters. This file is divided into sections, with each section starting with the line [*section-name*]. Some of the sections are used by all client applications, and others are used exclusively by a particular application. All available sections and configuration parameters are listed in Table 3. Parameters of the *sharedsrv* section, used by the Shared Cataloguing server are described in Table 2 and are omitted here.

Parameter	**Description**
Section: *general*	
locale	Locale to be used by client applications. Valid values are two-letter language abbreviations listed in the *ISO 639 Language Codes* [5].
Section: *database*	
type	The type of the database. For example, Oracle database is denoted by *oracle*, while SAP DB is denoted by *sapdb*.
driver	The name of the Java class representing the appropriate JDBC database driver. For example, Oracle database uses a driver named *oracle.jdbc.driver.OracleDriver*, while SAP DB database uses a driver named *com.sap.dbtech.jdbc.DriverSapDB*.
url	The JDBC address of the database used. The syntax of this URL depends on the type of the database used.
username	The username used for accessing the database.
password	The password used for accessing the database.
Section: *validation*	
validatorClass	The name of the Java class implementing record validation.
Section: *printing*	
print.command	The operating system command used for printing HTML pages.
browser.path	The absolute path to the web browser program used for printing HTML pages.
Section: *backup*	
destdir	Destination directory for backup.
Section: *editor*	

pingTime	Time period between server availability checks.
appServer	The type of network protocol for accessing the shared cataloguing server. The only allowed option is RMI_SERVER.
proxyHost	Address of the HTTP proxy host (if proxy is used for accessing external web services).
proxyPort	Address of the HTTP proxy port (if proxy is used for accessing external web services).
Section: *export*	
confirm	If 0, the Import/Export application will not ask for confirmation of invoked import/export operations.

Table 3. The client application set configuration parameters

6. Importing and Exporting Data

The BISIS Import/Export client application provides means for importing and exporting data from/to a BISIS database. There are four main database segments that can be imported or exported: (1) UNIMARC records, (2) librarian environment, (3) UNIMARC configuration, and (4) circulation data. These four segments are listed as items in the application's main menu presented in Fig. 4.

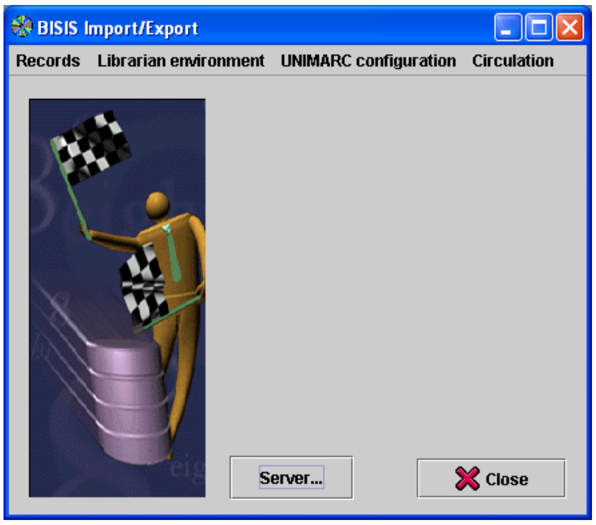


Figure 4. The main menu of BISIS Import/Export

Each of the main menu items provides options for import and export of the given sigment of the database. Export and import use a common text file format. The librarian environment, UNIMARC configuration and circulation segments are exported following the underlying relational database structure and are not intended to be shared among different library information systems. The file format for those segments uses Unicode UTF-8 encoding [6] and delimiter characters for identifying data rows and columns.

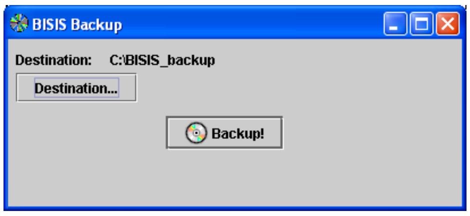
UNIMARC records are exported in a YUMARC-compliant format, assuming each record spans a single line in a text file. Line delimiters can be either a LF character or a CR/LF character combination. Details of file encoding are presented in [7].

7. Creating Backup Copies

Creating a backup copy of the BISIS database can be achieved in two ways. First, an administration tool for the underlying relational database used by the BISIS system may be used. This solution does not provide cross-database data availability since most backup utilities provided by database vendors use a proprietary file format. Second, a dedicated backup application for the BISIS system can be used. The format of the backup files is the same as for the BISIS Import/Export application. Hence, files produced by the BISIS Backup utility can be used in a BISIS Import/Export session. The BISIS Backup application reads all data from the BISIS database and stores it in a specific directory structure. The default destination directory for backup files is specified in the *config.ini* file, under *backup* section (see Table 3). Each backup session produces a directory structure, listed in Table 4, under the chosen backup directory. The BISIS Backup application main window is presented in Fig. 5. Clicking the *Backup!* button invokes the process of creating a backup copy. Destination directory can be chosen by clicking the *Destination...* button.

File name	**Description**
BISIS-yyyy-mm-dd	The name of the subdirectory containing backup files. Fields yyyy-mm-dd are determined by the current date.
records-yyyy-mm-dd-hh-min.dat	The file with UNIMARC records.
env-yyyy-mm-dd-hh-min.dat	The file with librarian environment data.
unimarc-yyyy-mm-dd-hh-min.dat	The file with UNIMARC configuration data.
circ-yyyy-mm-dd-hh-min.dat	The file with circulation data.

Table 4. Files produced by the BISIS Backup utility



Destination: C:\BISIS_backup

Figure 5. The main window of BISIS Backup utility

8. Conclusions

The BISIS system, following a three-tier internet application architecture, has a number of installation and configuration parameters. Some of these parameters need to be configured during system installation, and the others can be changed during the exploitation of the system.

Configuring servers and main client parameters is a part of the system installation task. All configuration parameters are stored in text files that can be edited with usual text editors. Reinstallation of program code in case of software updates does not affect system configuration because it is stored separately from the program code. Text files have no properties specific to a certain operating system, and can be easily exchanged among different platforms.

Importing and exporting data from a BISIS database uses text files thus facilitating later data editing and consumption. There is a special backup utility that uses the same file format as the Import/Export application providing complete data backup. UNIMARC records exported from a BISIS database are stored in a standard text file format and can be easily used in other information systems.

References

[1] D. Surla and Z. Konjović, eds. *Distributed Library Information System BISIS*. Group for Information Technologies, Novi Sad, 2004. (in Serbian).

[2] D. Coward. *Java Servlet Specification Version 2.3*, Sun Microsystems, 2001. http://java.sun.com/j2ee

[3] M. Roth, E. Pelegrí-Llopart. *JavaServer Pages Specification Version 2.0*. Sun Microsystems, 2003. http://java.sun.com/j2ee

[4] *Jakarta Tomcat*. http://jakarta.apache.org/tomcat

[5] ISO 639 Language Codes. http://www.w3.org/WAI/ER/IG/ert/iso639.htm

[6] *The Unicode Standard, Version 4.0.1*. Unicode Consortium. http://www.unicode.org/versions/Unicode4.0.1/

[7] M. Vidaković, B. Milosavljević. *Internationalisation and Localisation in the BISIS System*. In D. Surla, Z. Konjović, eds., *Distributed Library Information System BISIS*, Group for Information Technologies, Novi Sad, 2004. (in Serbian).

Internationalisation and Localisation of the Software System BISIS

Milan Vidaković, Branko Milosavljević
Faculty of Technical Sciences, University of Novi Sad
{minja,mbranko}@uns.ns.ac.yu

Abstract. Internationalization and localization means that programs should be made in a way that they are region and language independent. This means that existing programs should be modified and new programs should be made in such manner that adaptation to some particular region and language should be done without intervention on source code. The librarian information system BISIS enables equivalent usage of all available languages supported by the operating system (both fonts and keyboard layouts). This is due to the Unicode support present in the Java programming language. On the other hand, for the internal text representation UNIMARC standard is used. This standard enables an efficient conversion of different letters and a joint Cyrillic and Latin search. As a final result, the BISIS system enables the analogous use of both Cyrillic and Latin for input and display, and for defining query text, where both Cyrillic and Latin queries will produce the same result.

1. Introduction

Internationalisation and localisation [1] are the terms denoting writing software applications so that they do not depend on a particular country and language and that they are easy to adjust to a concrete state or language. Internationalisation means that an application is made in such a manner that it can be easily adjusted to any language without intervention on a program code. Localisation is the process of adjusting an application to a specific language and geographic area (locale). Locale is a geographic or a politically specific area with a unique language.

The BISIS library information system supports internationalisation and localisation. Both concepts have been implemented using the *Unicode* standard [2] and multilanguage features of the Java programming language.

BISIS localisation comprises of three segments:

- database localisation,
- librarian applications localisation and

- web site localisation.

Database localisation depends on the *Unicode* support [2] that is implemented in both RDBMS and Java database driver [3]. Database localisation can be performed in two ways:

- by translating database content into a specific language. This approach means that there is only one language in the database,
- by database schema modification so that it can hold multiple languages.

First approach does not need either a database schema modification or a programme modification. However, it is necessary to have a localised version of the database for each language. Also, it is not possible to use multiple languages simultaneously. Second approach makes database and application development more complicated, since it is necessary to modify both the database schema and the application. The BISIS system uses the first approach.

Librarian applications localisation and web site localisation use standard Java localisation methods which will be described in the section 3.

Besides internationalisation and localisation, the BISIS system equally handles both Latin and Cyrillic letters. If a record is entered in the Cyrillic letters, Latin queries will find it despite different alphabets, and vice versa.

2. Multilanguage Text Handling

2.1. Code Pages

There are many code page standards for the text representation. Code page standards are defined either by the standardisation committees like the ISO (*International Organization for Standardization*), or by the companies like *Microsoft*. ISO code page standards are members of the ISO 8859 [4,5] standards group and are widely used over the Internet. These are 8-bit single byte fixed width code pages. *Microsoft* company did not accept the ISO standards. Instead, *Microsoft* defined its own standards that slightly differ from the ISO standards. Some of the *Microsoft* code page standards are CP1250 (Central and East European Latin) and CP1251 (Cyrillic) [6].

There are several problems in multilanguage text exchange. First of all, there is a lack of support for all the standards on all platforms: on *Microsoft* platforms there is a code pages support defined by the *Microsoft*, while on the *Unix/Linux* operating systems ISO code page standards are supported. Besides, there is a lack of the appropriate fonts that would display the text using the corresponding code page.

The *Unicode* standard [2], which is defined by the *Unicode* consortium and accepted as the ISO 10646 standard, defines a completely different approach: it defines a unique code for every character regardless of the hardware/software platform, application or language. It uses 16 bits for each character having 2^{16} different characters which could be enough for all languages in use. There are several *Unicode* representations. UTF-8 and UTF-16 are most frequently used representations. Each UTF (*Unicode Transformation Format*) representation represents an algorithm for mapping a *Unicode* value into an array of bytes. Inverse mapping is also defined.

The UTF-8 main feature is that it is a single byte variable width format. This means that different characters take one, two or more bytes in the UTF-8 representation. It is designed so that characters from the ASCII character set have the same code in the UTF-8 representation. This simplifies the use of *Unicode* in older systems which are not designed for *Unicode*. This is the main reason why UTF-8 is mostly used on the Internet for the HTML [7] and XML [8] documents representation.

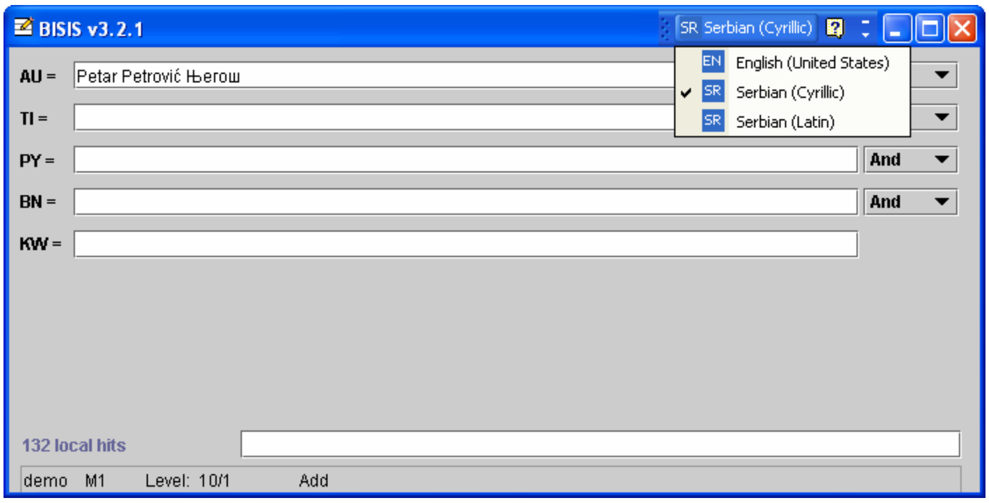
The UTF-16 format is a double byte fixed width format. Each character is represented using two bytes. This way, it is possible to differentiate between approximately 63000 characters, while using the surrogate mechanism provides support for approximately one million characters. The UTF-16 is used for the internal *Unicode* representation in *Java* and *.NET* frameworks.

The librarian information system BISIS uses the UNIMARC [9] standard for description and storage of librarian records. This standard is an international librarian standard and has many national variations like YUMARC [10, 11]. This standard also defines its own code page that is used for the multilanguage text representation.

This code page is a combination of several ISO standards. The main concept of this code page is the *current language* characteristic. This means that one code can represent several different characters, depending on the current language. The current language is represented by a single byte in a text. If there is no current language mark present at the beginning of the text, Latin is set as a default. The YUMARC standard supports Cyrillic, Latin, Greek and Old Cyrillic letters. The UNIMARC to *Unicode* mapping and vice versa is implemented in the BISIS system.

2.2. Displaying and Entering Multilanguage Text

The BISIS system is an Internet/Intranet application. Intranet users are mostly librarians who process the bibliographical material, i.e. create YUMARC records. For this purpose a special application is developed. The librarian application is a Java GUI application which supports *Unicode*. Java framework uses standard system-based mechanisms for entering and displaying *Unicode* characters. Figure 1 displays the main librarian application window with search text entered in two different languages using the system-based keyboard switching mechanism.



Petar Petrović Његош

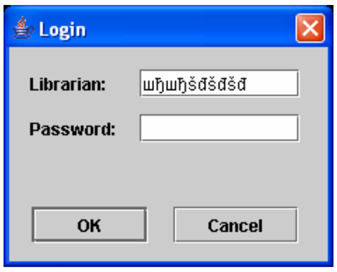
Figure 1. Multilanguage text support in the BISIS applications

Java applications use standard *TrueType* fonts (*Arial* on *Windows* platforms, *Lucida Sans* on *Linux*, etc.) for text display. These fonts support *Unicode* standard. Instead of these fonts, any other *Unicode TrueType* or *OpenType* font can be used.

3. Internationalisation and Localisation in the Java Programming Language

3.1. Unicode Implementation in GUI and Web Applications

Java GUI applications can be easily internationalised and localised because Java platform supports the *Unicode* standard. Moreover, the Java platform enables the *Unicode* characters input from the keyboard. Input is supported by the system keyboard switcher and supports all available languages in the system. Displaying the *Unicode* characters depends on the *Unicode* font implementation. Figure 2 displays the *Unicode* characters input and displaying.


Login

Librarian: шђшђšđšđšđ

Password:

Figure 2. Unicode character input and display

Internationalisation of web applications depends on the correct input and display of the *Unicode* characters. Since all interaction between the web application and a user is done using a web browser, there is a possibility of having incorrect input and display of characters.

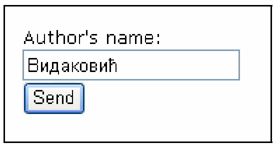
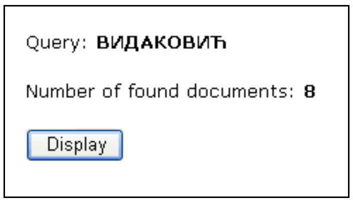

Author's name: Видаковић

Figure 3. Web form *Unicode* characters input


Query: **ВИДАКОВИЋ**

Number of found documents: **8**

Figure 4. Web page *Unicode* characters display

Figures 3 and 4 display the input and the display of *Unicode* characters in the BISIS web application. It is necessary to have the correct *Unicode* implementation in the web browser. At this moment, following browsers support *Unicode* entirely: *Internet Explorer*, *Netscape Navigator*, *Mozilla* and *Opera*.

3.2. Internationalisation and Localisation Classes in the Java Programming Language

The Java programming language supports internationalisation and localisation of the arbitrary programs [12]. There are specialised classes for the internationalisation and localisation in this programming language. Java also supports

Unicode standard entirely, which makes it an appropriate programming language for multilanguage environment.

Internationalisation and localisation means translating all text messages which appear in the programme into a specified language. Text messages are stored in files whose file names comply with the *Java Resource Bundle* mechanism. The main file containing messages appearing if there is no supporting locale is *Messages.properties*. The default language is English. Localised files have file names of the following form: *Messages_<locale>.properties*, where <*locale*> is a two-letters abbreviation of the geographic region being localised. All message files must be in the same application directory. Table 1 lists file names for the corresponding languages.

File	**Language**
Messages_en.properties	English
Messages_sr.properties	Serbian
Messages_mk.properties	Macedonian
Messages_de.properties	German
Messages_gr.properties	Greek
Messages.properties	default (English)

Table 1. File names for corresponding languages

The BISIS system consists of several applications. Each of them has its own set of localised message files. Applications are distributed as executable JAR files which contain the executable code together with message files.

Entering text into message files must be consistent to the Java 2 API rules. All non-ASCII characters are placed as *Unicode* values in the following textual form: \u*xxxx*, where *xxxx* is a four-digit hexadecimal *Unicode* value.

Figures 1 and 2 display a librarian application window localised for the Serbian and English languages.

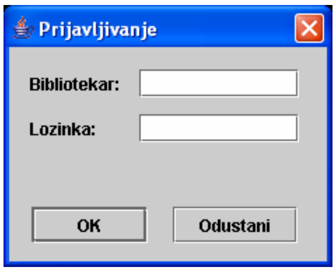


Figure 5. Login screen localised for the Serbian language

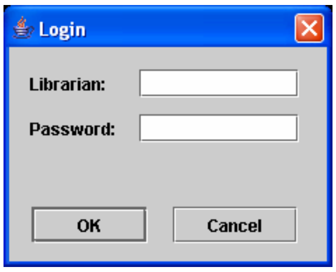


Figure 6. Login screen localised for the English language

4. Conclusion

The librarian information system BISIS enables equivalent usage of all available languages supported by the operating system (both fonts and keyboard layouts). This is due to the *Unicode* support present in the Java programming language. All interaction with the user is done in accordance with this standard, regardless of the type of the application used (a GUI application or a web application). On the other hand, for the internal text representation UNIMARC standard is used. This standard enables an efficient conversion of different letters and a joint Cyrillic and Latin search. As a final result, the BISIS system enables the analogous use of both Cyrillic and Latin for input and display, and for defining query text, where both Cyrillic and Latin queries will produce the same result.

Suggested localisation mechanism of the Java applications which was implemented is not designed with the express purpose for use in this system. It is generally applicable to all GUI and web Java applications.

References

[1] *Java ^{TM} 2 SDK Internationalization Overview*, http://java.sun.com/j2se/1.3/docs/guide/intl/index.html.

[2] Unicode Consortium Homepage, http://www.unicode.org.

[3] J. Ellis, L. Ho, M. Fisher. *JDBC 3.0 Specification*. Sun Microsystems, 2001. http://java.sun.com/j2se/

[4] ISO/IEC 8859-2:1987, Information Processing – 8-bit Single Byte Coded Character Sets – Part 2: Latin Alphabet No. 2, International Organization for Standards, http://www.iso.ch

[5] ISO/IEC 8859-5:1988, Information Processing – 8-bit Single Byte Coded Character Sets – Part 5: Latin/Cyrillic Alphabet, International Organization for Standards, http://www.iso.ch

[6] Microsoft Corporation, http://www.microsoft.com

[7] D. Raggett, A. Le Hors, I. Jacobs. *HTML 4.01 Specification*, W3C Consortium, 1999. http://www.w3.org/TR/html4/

[8] T. Bray, J. Paoli, C.M. Sperberg-McQueen, E. Maler. Extensible Markup Language (XML) 1.0 (Second Edition). W3C Recommendation, 2000. http://www.w3.org/TR/REC-xml

[9] *UNIMARC Manual: Bibliographic format*, *International Federation of Library Association and Institutions*, IFLA Universal Bibliografic Control and International MARC Programme, New Providence, London 1994.

[10] *COMARC/R Format*, COBISS, Univerzitetni institut informacijskih znanosti, IZUM, Maribor 1992.

[11] B. Lazarević, redaktor. *Formiranje i pretraživanje baza podataka u Sistemu naučnih i tehnoloških informacija Srbije*, Ministarstvo za nauku i tehnologiju Republike Srbije, Beograd, 1996.

[12] Vidaković, M., Milosavljević, B., Konjović Z., *Internartionalization of Java Applications*, Proceedings of the YuInfo 2003 (CD), Kopaonik 2003 (in Serbian).

Network Infrastructure Design for a BISIS System in an Inter-Library Scenario

Milan Kerac, Ivan Nejgebauer
University of Novi Sad
{kerac,ian}@uns.ns.ac.yu

Abstract. This paper describes the physical and logical architecture of the computer network which supports the library information system BISIS. It is assumed that the BISIS systems in the network are part of a larger inter-library interconnection over the global Internet infrastructure. The paper also discusses the design of a basic access-control policy to the individual modules and services of the BISIS system.

1. Introduction

Operation of a distributed information system based on the client-server model depends on the underlying network infrastructure. Current distributed information system are predominantly based on the TCP/IP protocol suite[2]. This paper proposes the architecture of an interuniversity library network based on the BISIS system. BISIS is a distributed information system based on the client-server model, and uses the TCP/IP protocol suite. Network infrastructure encompasses the physical, data link, network and transport layers of the TCP/IP protocol suite.

BISIS installations are currently deployed in Merseburg, Novi Sad, Skopje and Thessaloniki. The installation in Novi Sad is internally more complex than the others since it consists of multiple networked library servers. The network infrastructure example presented in the paper describes the more complex setup; single-server setups can be derived by simplifying the described infrastructure.

2. Initial premises

The example describes the network infrastructure supporting two libraries within the BISIS system installation in Novi Sad. It is assumed that:

- Both BISIS servers provide the Web search interface to global Internet users.
- Both servers allow mutual remote cataloguing.
- Both servers provide entry retrieval to each other.
- The server of Library 1 allows entry retrieval from a limited set of external libraries.

- The server of Library 2 does not allow entry retrieval from external libraries.
- The staff of Library 1 and Library 2 can directly access only the RDBMS of their respective servers.

3. Physical architecture

The physical architecture of the system is shown in Figure 1.

3.1. Library 1

Library 1 is situated in the main building of the organization it belongs to. Dual-socket network outlets for connecting the workstations of Library 1 staff to the Library 1 BISIS server are installed at the premises of Library 1. All network outlets are fitted with Category 5e[1] compliant sockets. The sockets are the endpoints of the communication links originating from the local concentrator LC-5-1. Physically, the links are Category 5e Unshielded Twisted Pair (UTP) cables placed in wall-mounted PVC cable conduits. The originating point of the links is a patch panel in LC-5-1, labeled LC-5-1-PP-2. The patch panel has 20 Category 5e sockets and 2 SC multimode sockets. All end-user network outlets are connected through the patch panel and by patch cables to the ports of the switch LC-5-1-SW-1. The switch has 24 RJ-45 10/100 Mb/s Ethernet ports and one SC 1000 Mb/s multimode port.

The local concentrator LC-5-1 is further connected to the central distributor CD. That connection starts at the SC port of the switch LC-5-1-SW-1, via a multimode fiber patch cable to the SC socket of the patch panel LC-5-1-PP-2 and then through a secondary communications link. The link terminates at the SC socket of the patch panel LC-5-1-PP-2 and originates from the patch panel CD-PP-3, which has 12 SC multimode sockets. Physically the link is a fiber-optics cable with four multimode 50/125 um fibers, placed in PVC cable conduits. All CD-PP-3 sockets are connected to the central switch CD-SW-1 by the appropriate multimode fiber patch cables. CD-SW-1 has twelve SC 1000 Mb/s multimode ports and two RJ-45 1000 Mb/s (copper) ports.

3.2. Library 2

Library 2 is situated in a separate building. Dual-socket network outlets for connecting the workstations of Library 2 staff to the Library 1 BISIS server are installed at the premises of Library 2. All network outlets are fitted with Category 5e compliant sockets. The sockets are the endpoints of the communications links originating from the local concentrator LC-B-2-1-1. Physically, the links are Category 5e UTP cables placed in wall-mounted PVC cable conduits. The originating point of the links is a patch panel in LC-B-2-1-1, labeled LC-B-2-1-1-PP-1. The patch panel has 24 Category 5e sockets. All end-user network outlets are connected through the patch panel and by patch cables to the ports of the switch LC-B-2-1-1-SW-1. The switch has 24 RJ-45 10/100 Mb/s Ethernet ports and one RJ-45 1000 Mb/s Ethernet port.

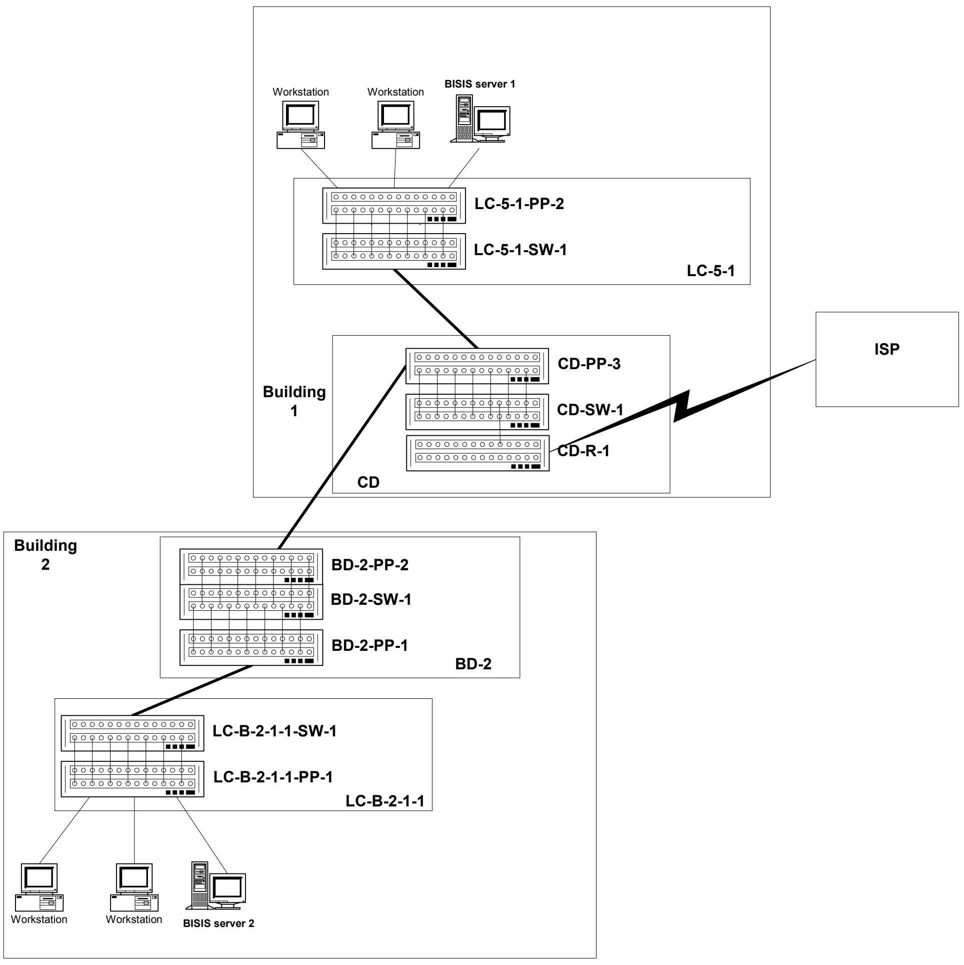


Figure 1. Physical architecture of the BISIS installation.

The local concentrator LC-B-2-1-1 is further connected to the building distributor BD-2. That connection starts at the 1000 Mb/s port of the switch LC-B-2-1-1-SW-1, via a patch cable to the patch panel LC-B-2-1-1-PP-1 and then through a secondary communications link. The link terminates at the patch panel LC-B-2-1-1-PP-1 and originates from the patch panel BD-2-PP-1, which has 24

Category 5e sockets. Physically the link is a Category 5e UTP cable placed in PVC cable conduits. The BD-2-PP-1 socket is connected to the switch BD-2-SW-1 by the appropriate patch cables. BD-2-SW-1 has ten RJ-45 1000 Mb/s Ethernet ports and two SC 1000 Mb/s multimode ports.

The building distributor BD-2 is connected to the central distributor CD. That connection starts at the SC port of the switch BD-2-SW-1, via a multimode fiber patch cable to the SC socket of the patch panel BD-2-PP-2 (20 Category 5e sockets and two SC multimode sockets) and then through a primary communications link. The link terminates at the SC socket of the patch panel BD-2-PP-2 and originates from the patch panel CD-PP-3. Physically the link is a fiber-optics cable with four multimode 50/125 um fibers, placed in an underground cable run between the buildings and in PVC cable conduits inside the buildings. The CD-PP-3 sockets is connected to the central switch CD-SW-1 by the appropriate multimode fiber patch cable.

3.3. Central Distributor

In addition to the devices which support the local network, the central distributor houses the router CD-R-1, which connects the local network to the Internet.

4. Logical Architecture

4.1. Public IP Address Space

Libraries 1 and 2 use a part of the public IP address space allocted to the organization to which they belong. The libraries have the public IP address range with the following parameters:

- Network identification is 147.91.179.32;
- Subnet mask is 255.255.255.248, i.e., mask length is 29;
- Broadcast address is 147.91.179.39.

Accordingly, public IP address allocation is as follows:

- The public IP address of the Library 1 server is 147.91.179.33/29;
- The public IP address of the Library 2 server is 147.91.179.34/29;
- Other public IP addresses from the allocated range are assigned to the network devices which support the logical network segment using that address range.

4.2. Private IP Address Space

Libraries 1 and 2 use a part of the private IP address space[3] used by the organization to which they belong. Private IP address allocation for the libraries is as follows:

- Library 1 uses the range 192.168.171.0/24;
- Library 2 uses the range 192.168.172.0/24;
- Public reading rooms use the range 192.168.111.0/24;
- Administrative staff uses the range 192.168.254.0/24;

4.3. DNS Naming

DNS names of the devices (servers and workstations) belonging to both libraries are hierarchical subordinates of the organization.ns.ac.yu name space.

Names used by Library 1 belong to the lib1.organization.ns.ac.yu name space (www.lib1.organization.ns.ac.yu, catalog.lib1.organization.ns.ac.yu).

Names used by Library 2 belong to the lib2.organization.ns.ac.yu name space (www.lib2.organization.ns.ac.yu, catalog.lib2.organization.ns.ac.yu).

4.4. Virtual Local Area Networks (VLANs)

Implementation of the described logical architecture requires the setup of a number of VLANs:

- Library 1 staff VLAN. Workstations of the Library 1 staff and the BISIS server network interface which they access belong to this VLAN. The VLAN is labeled L1VLAN and is associated with the IP address range 192.168.171.0/24.
- Library 2 staff VLAN. Workstations of the Library 2 staff and the BISIS server network interface which they access belong to this VLAN. The VLAN is labeled L2VLAN and is associated with the IP address range 192.168.172.0/24.
- Public reading rooms VLAN. Workstations installed in the public reading rooms of both libraries belong to this VLAN. The VLAN is labeld RRVLAN and is associated with the IP address range 192.168.111.0/24.
- Global access VLAN. BISIS server network interfaces for global Internet access belong to this VLAN. The VLAN is labeled GVLAN and is associated with the public IP address range 147.91.172.32/29.
- Administrative staff VLAN. Workstations of administrative staff belong to this VLAN. The VLAN is labeled ADVLAN and is associated with the IP address range 192.168.254.0/24.

4.5. Traffic Flow Configuration

The local area network switches are configured with the following VLANs:

- Switch LC-5-1-SW-1: L1VLAN, GVLAN and ADVLAN.
- Switch LC-B-2-1-1-SW-1: L2VLAN, GVLAN and ADVLAN.
- Switch BD-2-SW-1: L2VLAN, GVLAN and ADVLAN.
- Switch CD-SW-1: L1VLAN, L2VLAN, RRVLAN, GVLAN and ADVLAN.

Every switch port belongs to one of the defined VLANs, except for switch trunking ports and BISIS server ports. These ports use the IEEE 802.1q trunking protocol to tag each transferred frame with its VLAN identifier. The IEEE 802.1q protocol is supported on all switches, as well as the BISIS servers.

The switch CD-SW-1 is configured to allow bidirectional traffic flow between the private IP networks of Libraries 1 and 2 and other relevant private IP networks within the organization. The switch CD-SW-1 and the router CD-R-1 are configured to allow global Internet access to the public IP addresses assigned to the public network interfaces of the BISIS servers. Traffic between the private and the public IP addresses is not allowed.

4.6. BISIS Network Services Configuration

The BISIS servers of both libraries support the IEEE 802.1q trunking protocol. Active network interfaces and their parameters are summarized in Table 1.

BISIS server	**Network interface label**	**VLAN associated with the IP address of the interface**
Library 1	eth0.1	GVLAN
	eth0.2	L1VLAN
Library 2	eth0.1	GVLAN
	eth0.2	L2VLAN

Table 1. Network interfaces of the BISIS system.

Network services available on the BISIS servers of Libraries 1 and 2 are described in Table 2.

BISIS server	**Service description**	**Protocol**	**Port**	**VLAN to which the service IP address belongs**
Library 1	Web interface	TCP	80	GVLAN
	Web interface	TCP	80	L1VLAN
	RDBMS	TCP	7200	L1VLAN

	RMI	TCP	1099	GVLAN
	RMI	TCP	1099	L1VLAN
	SSH	TCP	22	L1VLAN
Library 2	Web interface	TCP	80	GVLAN
	Web interface	TCP	80	L2VLAN
	RDBMS	TCP	7200	L2VLAN
	RMI	TCP	1099	L2VLAN
	SSH	TCP	22	L2VLAN

Table 2. Network services of the BISIS system.

4.7. Access Control

Access to the network services and modules of the BISIS system is controlled at several levels.

- Division of the local network into VLANs creates a separate broadcast domain for each VLAN and restricts IP communication between the VLANs according to the rules imposed by traffic flow control.
- Packet filtering on the devices CD-R-1 and CD-SW-1, as well as the BISIS servers, further restricts the types of traffic flow between the network components.
- CD-R-1, physical interface to the Internet provider, inbound. Unrestricted access is allowed to the TCP port 80 and the public IP addresses of the BISIS server. (To prevent spoofing, inbound traffic with source IP addresses belonging to the organization's public IP address space is disallowed.) Access to the TCP port 1099 and the public IP addresses of the BISIS server is allowed for the traffic originating from the IP addresses of BISIS installations at Merseburg, Skopje and Thessaloniki. Other traffic to the public IP addresses of the BISIS servers is not allowed.
- CD-SW-1, GVLAN virtual interface, outbound. Unrestricted access is allowed to the TCP port 80 and the public IP addresses of the BISIS server, along with the anti-spoofing restriction described in the previous paragraph. Access to the TCP port 1099 and the public IP addresses of the BISIS server is allowed for the traffic originating from the IP addresses of BISIS installations at Merseburg, Skopje and Thessaloniki. In accordance with the organization's network policy, further rules not in conflict with the preceeding two may also be defined. Other traffic to the public IP addresses of the BISIS servers is not allowed.
- CD-SW-1, L1VLAN virtual interface, outbound. Access is allowed from the private IP addresses of ADVLAN to the private IP address of the Library 1 BISIS server and TCP ports 22, 80, 1099, and 7200. Access is allowed from

the private IP addresses of L2VLAN to the private IP address of the Library 1 BISIS server and TCP ports 80 and 1099. Access is allowed from the private IP addresses of RRVLAN and other private networks (as determined by the organization's internal traffic flow policy) to the private IP address of the Library 1 BISIS server and TCP port 80. Further rules not in conflict with the preceeding ones may also be defined. Other traffic to the private IP address of the Library 1 BISIS server is not allowed.

- CD-SW-1, L2VLAN virtual interface, outbound. Access is allowed from the private IP addresses of ADVLAN to the private IP address of the Library 2 BISIS server and TCP ports 22, 80, 1099, and 7200. Access is allowed from the private IP addresses of L1VLAN to the private IP address of the Library 2 BISIS server and TCP ports 80 and 1099. Access is allowed from the private IP addresses of RRVLAN and other private networks (as determined by the organization's internal traffic flow policy) to the private IP address of the Library 2 BISIS server and TCP port 80. Further rules not in conflict with the preceeding ones may also be defined. Other traffic to the private IP address of the Library 2 BISIS server is not allowed.
- Library 1 BISIS server, L1VLAN virtual interface, inbound. Access is allowed from the librarians' workstations in L1VLAN to the TCP ports 80 and 7200. Other workstations in L1VLAN can access only the TCP port 80. Other traffic from the L1VLAN workstations is not allowed.
- Library 2 BISIS server, L2VLAN virtual interface, inbound. Access is allowed from the librarians' workstations in L2VLAN to the TCP ports 80 and 7200. Other workstations in L2VLAN can access only the TCP port 80. Other traffic from the L2VLAN workstations is not allowed.
- User authentication is performed for all network services of the BISIS servers, except the public Web search service reachable through the TCP port 80.

5. Conclusion

The described physical and logical architecture supports the deployment of the library information system. Configuration parameters provide high system availability and performance, while access control to the system components guards the system from unauthorized use, further enhancing the stability of the library information system. Introduction of new services and user categories does not change the baseline configuration and requires just the amendment of the existing rules, which enables unhindered further development of the whole system.

References

[1] ANSI/TIA/EIA-568-A, Commercial Building Telecommunications Cabling Standard.

[2] Braden, R. (Ed.), *Requirements for Internet Hosts - Communication Layers*, RFC 1122, October 1989.

[3] Rekhter, Y. et al., *Address Allocation for Private Internets*, RFC 1918, February 1996.

Software Architecture of the Networked Digital Library of Theses and Dissertations

Srđan Komazec, Goran Sladić, Dušan Okanović, Zvezdan Protić
Faculty of Technical Sciences, University of Novi Sad
{ksrki,sladicg,oki,zvezda_n}@uns.ns.ac.yu

Abstract. This paper presents software architecture of Digital Library of Theses and Dissertations (DIGLIB) system. DIGLIB is composed of web based networked services providing functionality such as full text publication of theses and dissertations, searching based on various criteria, and cooperative work on publishing process between librarians and authors. It's software architecture is based on an n-tier application stack and relational database employing J2EE technology and SAP DB. Design and implementation concepts used in DIGLIB system follow design patterns such as Command, Sequence blocks, Model-View-Controller, Object pool, etc which provide us with more flexible, reusable, and scalable solution which is easy to maintain and with high performance.

1. Introduction

In 2003 Provincial Secretariat for Science and Technological Development of Autonomous Province of Vojvodina initiated a project "Networked Digital Library of Theses and Dissertations" [1]. The goal was to develop web based network service that would allow faculties to publish their candidates' papers and make them accessible over internet. Data can be accessed through user and system interface or over international network NDLTD (Networked Digital Library of Theses and Dissertations) [2] using PMH protocol [3]. This paper presents general software architecture of DIGLIB, design and structure of the implemented software components and user interface and implementation issues (J2EE platform [4]).

2. Digital Library Software Architecture

There are three logical layers of this system:

- Data layer,
- Process layer, and
- Interface layer.

The goal of this partition is a semantic separation of internal system functions. This allows us to use all the advantages of multilayer architecture (performance, flexibility, maintenance, reusability and scalability) [5].

The data layer allows efficient information storage, update and retrieval. The process layer simplifies design and implementation of basic processes for data manipulation and their integration into more complex processes. The interface layer uses the other two layers to present information to users, to accept their requests, and generate corresponding actions (by activating proper processes). The aim of the interface layer is also to connect DIGLIB system with other systems. Every layer uses transaction and security services.

The DIGLIB layered structure is shown in Figure 1.

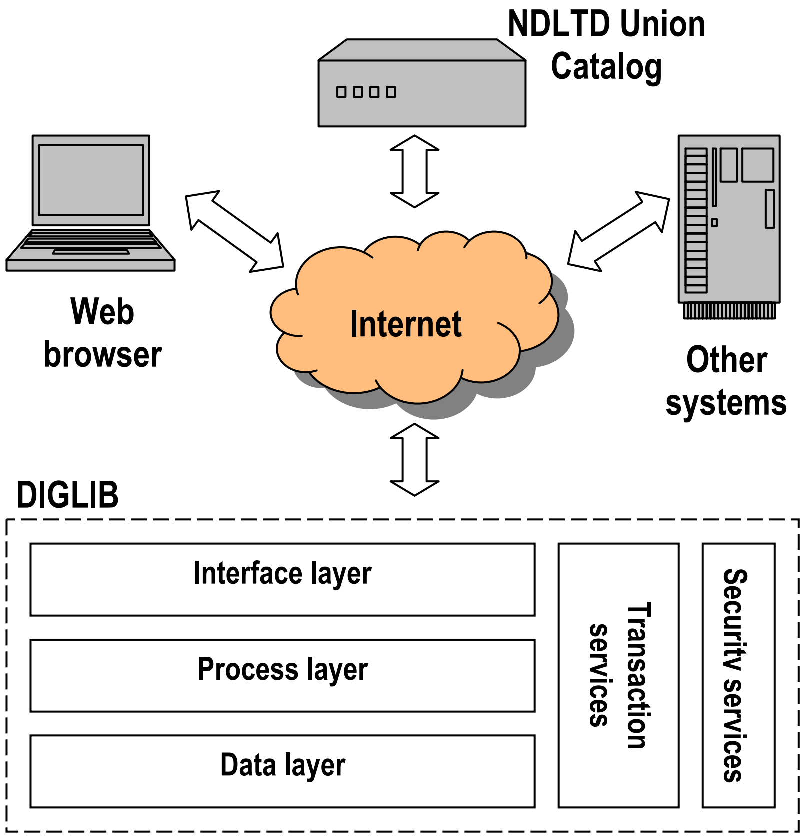


Figure 1. The DIGLIB layered structure

2.1. The Data Layer

The data layer of this system has two parts: the main metadata database and an index database (to provide for efficient metadata retrieval).

The main database has 39 entities grouped in five categories:

- users (system administrators, faculty administrators, librarians, authors),
- institutions (universities, faculties, libraries),
- documents (language independent and language dependent data – title, key words, abstract and extended abstract),
- defend board members, and
- catalogues (language, country, record type, document type, title, content code, scientific field, scientific discipline).

The main database design assumes the use of an object-relational mapping tool. The goal was to create such an abstraction that would not depend on the target relational database management system (RDBMS) and facilitate data manipulation on the level of the object-oriented model, leaving all storing, updating, transaction and security tasks to services provided by the application server.

The index database was designed and implemented using a text server for UNIMARC records [6]. This database can be accesed directly using relational database acces APIs or using web services components.

2.2. The Process Layer

The process layer implements the following tasks:

- user data management,
- institution data management,
- metadata workflow management,
- document publishing management,
- user interface management, and
- system interface management.

These processes are of various types: synchronous or asynchronous, and user-oriented or software-oriented. Therefore, different software solutions were required. User-oriented processes were designed using the command design pattern [7] in order to separate logical segments, simplify maintenance, increase reliability and reusability of the code. Indexing of bibliographic metadata is an asynchronous task performed in the background. The implementation of this process uses asynchronous message-driven components. Web services were used for communication with other systems [8].

2.2.1 Command Design Pattern

When a request is encapsulated into an object, it can be parameterized, queued and logged. The central part of this design pattern is an abstract class `Command` that is an interface for operation execution. The simplest form of this interface contains only an abstract method `Execute`. The command separates the object that invokes the operation from the one that knows how to perform it.

Commands are complete objects and we can use any object-oriented programming mechanism to manage them. Commands can be integrated into macro commands. New commands can be added without changing existing classes. Temporary storage of executed commands (command cache) can be implemented in order to increase performance of the system.

Most of the process layer logic was designed and implemented using the command pattern. One example is the instantiation of the command when server receives key documentation data. Command is then initialized and executed. The result is the creation of new persistent document metadata.

2.2.2. Message Oriented Middleware

Message passing is an alternative to remote procedure calls, which results in better performance (no need to block the client), reliability (guaranteed delivery), and multiple sender and multiple receiver communication. The data is exchanged using Message Oriented Middleware (MOM) as presented in Figure 2. The DIGLIB system provides services for guaranteed delivery, fault tolerance and load balancing of message receivers.

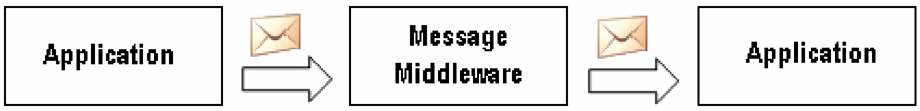


Figure 2. Message oriented middleware

The IndexerBean is a point-to-point software component that uses MOM infrastructure of the application server. It's task is to accept document metadata, perform indexing and prepare it for searching. The preparation of the data that is to be sent to the IndexerBean is performed in command whose method `Execute` is invoked through a web interface by a librarian. This way, miltiple librarians may asynchronously edit documents, while documents are indexed following the order of their arrival at the application server.

2.3. The Interface Layer

Concurrent work of multiple users is supported through the use of web-based user interface that facilitates deployment and maintenance. The design of the web application is driven by the Model-View-Controller (MVC) design pattern [9].

2.3.1. Model-View-Controller Design Pattern

Roots of MVC design pattern are in Smalltalk, where it was originaly used for mapping standard input, data processing and, user interface management. It has three logical components (Figure 3):

- *Model* – represents data and data manipulation rules.
- *View* – presents the contents of the model to the user. Data is accessed through the model component. The view components maintains consistency between the presentation and the model, when a change in data occurs. There may be multiple views for a single model.
- *Controller* – maps user actions performed within a view into actions on the model (process activation and/or model state change). The controller's response to actions is the activation of an appropriate view.

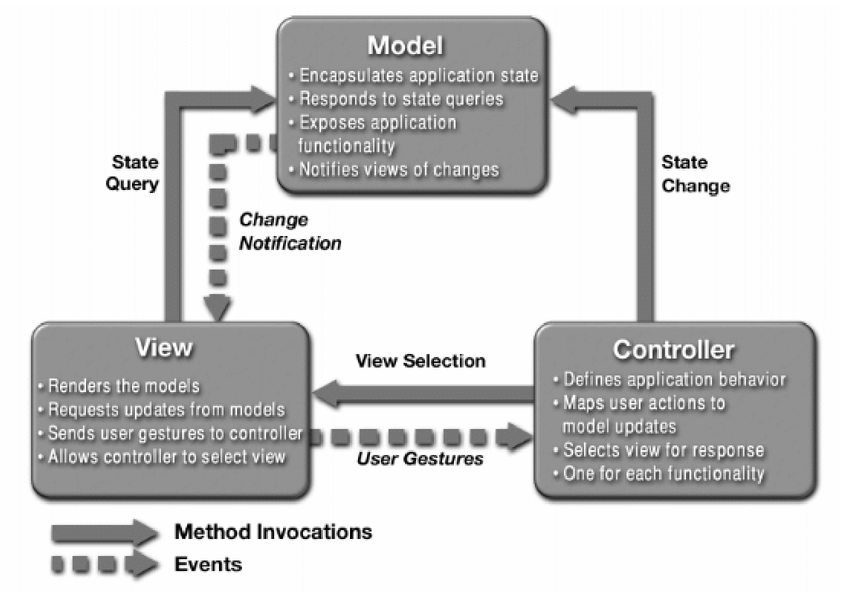

Model
• Encapsulates application state
• Responds to state queries
• Exposes application functionality
• Notifies views of changes
State Query
State Change
Change Notification
View
• Renders the models
• Requests updates from models
• Sends user gestures to controller
• Allows controller to select view
View Selection
User Gestures
Controller
• Defines application behavior
• Maps user actions to model updates
• Selects view for response
• One for each functionality
Method Invocations
Events

Figure 3. Model-View-Controller

The MVC pattern increases the flexiblity of model use and supports multiple types of users. On the other hand, the system complexity is increased as well. This pattern is combined with the Command pattern.

The MVC design pattern plays a central role in design and implementation of the user interface. The typical sequence of activities in a system implemented using the MVC pattern is as follows:

- A view (an HTML page) sends data to the server about the intended activity.
- The data is received by the controller, then validated and edited.
- The controller instantiates a command needed for the intended activity, sets command parameters and executes the command.
- Depending on the result of the command, the controller selects a view, sets the executed command in its scope and activates the selected view (the data is inserted in the HTML page).

2.4. Other design patterns used in this system

Other design patterns used in this system are Sequence Block [10] and Object Cache pattern [11].

2.4.1. Sequence Block Design Pattern

Sequence blocks generate sequences of numbers that may be used as primary keys with both BMP and CMP components [12]. One implementation of a sequence block is to build a component representing a sequence (i.e., storing numbers into database and incrementing them on each client request for the next value of the primary key). This solution induces poor performance, and scalability and complexity problems.

An alternative solution is to use the Session Bean [12] component in front of the Entity Bean component, taking blocks of numbers and caching them locally. This approach increases performance, scalability, simplicity, and reusability, but it can not guarantee a proper order (ascending or descending) of generated values, because some values can be lost when a Session Bean is removed by the application server or when the application server crashes.

The Sequence Block pattern solves the problem of primary key generation for all entities being a part of the data model. This solution is portable, i.e. RDBMS independent.

2.4.2. Object Cache Design Pattern

Object Cache pattern manages object reuse when their creation is time- and space-consuming or when the number of reusable objects is limited.

This pattern reduces object instantiation requirements, has better performace and lower memory requirements. On the other hand, cache size limitation can result in problems when object pool is exhausted because it requires new object creation or an existing object to be returned to the pool.

Object cache pattern is used to speed up access to the data model entities within commands in the system.

3. Digital Library Implementation

In order to decrease developement costs *open source* products were chosen. The operating system is Red Hat Linux, and database is SAP DB. The software platform is Java 2 Enterprise Edition, implemented by the JBoss application server. Jakarta Ant and XDoclet development tools were used for the implementation.

DIGLIB is a multilayer application developed using J2EE technology. Figure 4 presents its general architecture characteristics. The data layer comprises a SAP database [13] and Entity Bean components. CMP Entity Beans support database independence and facilitate handling of persistent entities. The process layer was implemented using various software components. Persistent entities were implemented using the Command pattern, whereas commands are JavaBeans. Servlets [14] are used as controllers and JSP pages [15] are used as views in the MVC pattern. Asynchronous message exchange (metadata indexing) was implemented using a Message Driven Bean [12] component. Web services were used for communication with other systems (mainly for metadata retrieval). Transaction and security management are provided by JBoss application server [16].

J2EE specification allows the creation of different clients, that communicate with the application server using standard protocols (JRMP[17], HTTP[18], or SOAP[19]).

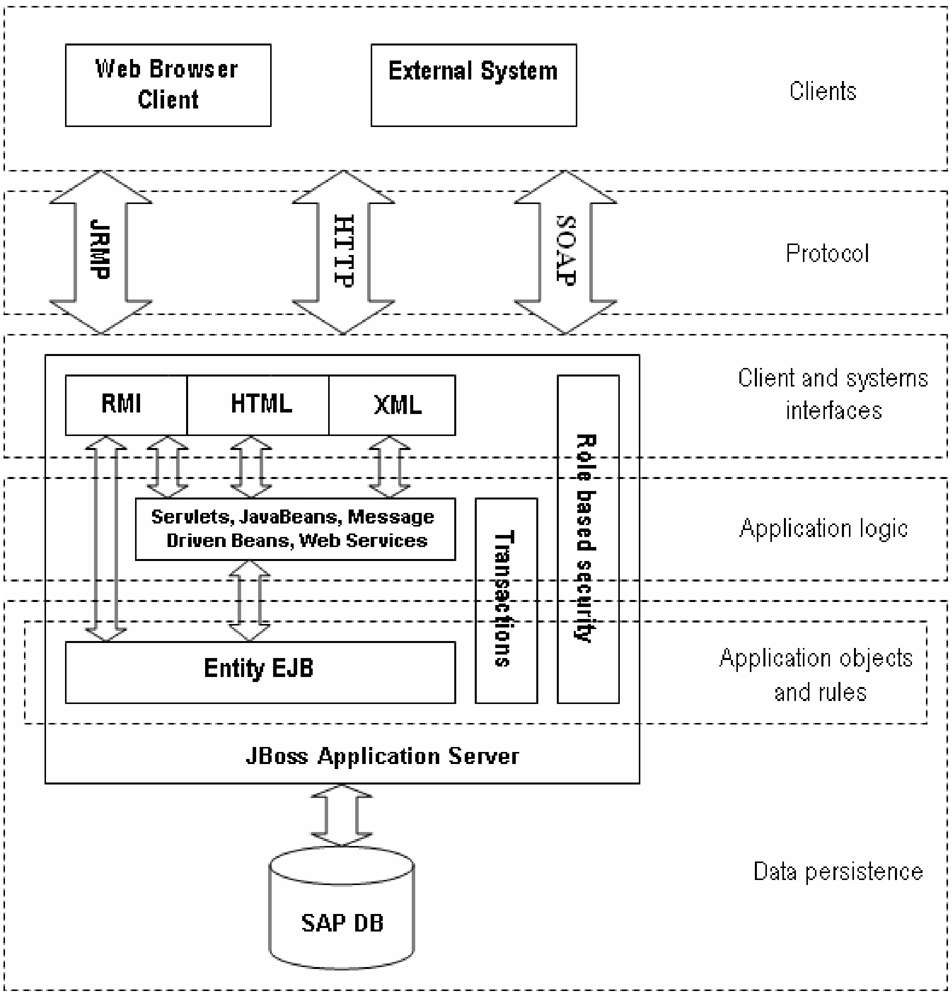


Figure 4. System's architecture considering the J2EE programming model

4. Conclusion

The DIGLIB system is a web-based, multi-layered software system with three different logical layers (data, process and interface layers). The data layer defines persistent entities of the system, that are mapped to the database using OR techniques. The process layer implements application logic managing the persistent entities. The structure and hierarchy of this layer are based both on design patterns (MVC, Command, Sequence Blocks, Object Pool) and web services providing a connection to other systems. The system was implemented using J2EE and open source components.

References

[1] *Project Network Digital Library of Doctor, Master and Diploma Theses* (project leader prof. dr. Dušan Surla), Provincial Secretariat for Science and Technological Development of Autonomous Province of Vojvodina, Novi Sad, 2003. http://diglib.ns.ac.yu

[2] *Networked Digital Library of Theses and Dissertations*. www.ndltd.org

[3] Zarić, M., Surla D., *Metadata Dissemination Using OAI-PMH*. International Conference on Distributed Library Information Systems, Ohrid, 2004.

[4] *Java 2 Platform, Enterprise Edition*. http://java.sun.com/j2ee

[5] *Three Tier Software Architectures*. http://www.sei.cmu.edu/str/descriptions/threetier.html

[6] Milosavljević, B., *Text Server for UNIMARC Records*, Master thesis, Faculty of Technical Sciences, University of Novi Sad, 1999. http://diglib.ns.ac.yu/ndltd/docs/set1/ndltd6/textsrv.pdf

[7] [7] Gamma, E., Helm, R., Johnson, R., Vlissides, J., *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, New York, 1994.

[8] Zarić, M., Milosavljević, B., *User Search in Digital Library of Theses and Dissertations of University of Novi Sad*, International Conference on Distributed Library Information Systems, Ohrid, 2004.

[9] Tate, B., *Bitter Java*, Manning Publications, Greenwich, 2002

[10] Marinescu, F., *EJB Design Patterns*: *Advanced Patterns, Processes and Idioms*. John Wiley and Sons, Inc., New York, 2002

[11] *Pattern Summaries: Object Pool*. http://www.developer.com/tech/article.php/626171

[12] *Enterprise Java Beans 2.0 Specification Final Release*. http://java.sun.com/products/ejb/docs.html

[13] *SAP DB*. www.sapdb.org

[14] *Java Servlet Technology*. http://java.sun.com/products/servlet

[15] *JavaServer Pages Technology*. http://java.sun.com/products/servlet

[16] *JBoss: Professional Open Source*. http://www.jboss.org

[17] *Java Remote Method Invocation (RMI).* http://java.sun.com/j2se/1.3/docs/guide/rmi/

[18] *HTTP – Hypertext Transfer Protocol*. http://www.w3.org/Protocols/

[19] *SOAP – Simple Object Access Protocol*. http://www.w3.org/TR/soap/

Document Processing in the Digital Library of Theses and Dissertations

Zvezdan Protić, Faculty of Technical Sciences, Novi Sad
zvezda_n@uns.ns.ac.yu

Abstract. The paper presents partial results of the project of the Networked Digital Library of Theses, Dissertations and Graduation Theses of the University of Novi Sad [1], specifically the processing of documents in the Digital Library of Theses and Dissertations. Short analysis of document structure is also given, along with a typical scenario for document creating and editing.

1. Introduction

The Digital Library of Theses and Dissertations (DIGLIB) was set up at the University of Novi Sad in 2003, with an express goal of creating a system to provide for storing and retrieving metadata and electronic versions of graduation theses, master theses and Ph.D. dissertations. The project was inspired by the NDLTD initiative conducted by the Virginia Tech University [2].

The DIGLIB is a web application which is easily accesible from any web-browser, and it is divided into two parts: BackOffice and FrontOffice. The BackOffice enables authors to store their theses and/or dissertations, while the FrontOffice contains a search engine which enables users to search through stored theses and dissertations by several criteria [3].

Generally, there are two types of system users, non-registered and registered. Registered users can play roles of authors, librarians, system administrators or faculty administrators. All registered users must log-in into the system in order to work with it. Non-registered users can only search through published documents.ž

2. Documents in the Digital Library of Theses and Dissertations

Metadata of a thesis or a dissertation, along with the corresponding electronic version of the paper, are called a *document*. Document metadata consist of two parts: language independent metadata and language dependent metadata. This division of metadata stems from the need for the digital library to be a multilingual application.

All document metadata not depending on the language, i.e. having the same meaning in every language the document is published in, are called language independent metadata. Metadata comprising this group are:

- document type,
- paper content,
- author,
- thesis supervisor metadata (title, name, surname),
- language of text,
- country and locality of publication,
- publication year,
- physical description,
- scientific area and scientific discipline,
- universal decimal classification,
- date of acceptance by scientific board,
- date of defending, and
- defended board members data.

Language dependent metadata must be entered for every language we want to publish the document in. Language dependent metadata are:

- title,
- keywords,
- abstract,
- extended abstract, and
- annotations.

The documents in the Digital Library of Theses and Dissertations can be in one of the two states: *unpublished* or *published*. Unpublished documents are being edited. After editing the document has been completed, it is published. The process of publishing means mapping metadata to the UNIMARC format and making metadata and electronic version of the thesis accessable through internet; the search is carried out by using the UNIMARC prefixes.

Published documents can be searched for in the FrontOffice.

3. Document Creating, Editing and Publishing

Document creating, editing and publishing is done in the BackOffice module. The Figure 1 shows the use-case of the BackOffice module.

The process of creating, editing and publishing documents is taken part in by librarians and authors, and this part of the use-case diagram will be presented in detail. Librarians can browse through the list of authors, register new authors, and create new documents for existing authors. Authors are in charge of editing their documents. Each author gets a user name and a password when applying and a document is created for him/her.

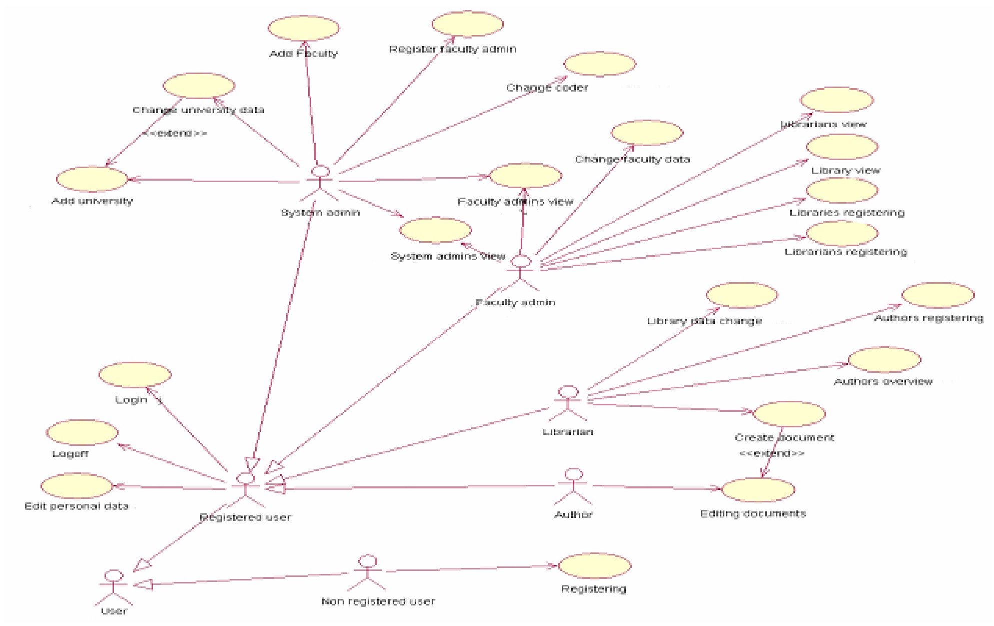


Figure 1. BackOffice use-case diagram

Document creation. The document editing process is started by a librarian, who checks if the author is already registered. If the author is registered, a new document is created for him/her as shown in the Figure 2. Required metadata for creating a new document are: document type, paper content and date of acceptance by scientific board. Otherwise, if the author is not yet registered, the librarian registers him/her on the system as shown in the Figure 3. Author's data consist of the following fields: name, surname, username, password, profession, address, city, phone, e-mail address. The new document for the author is then being created subsequently.

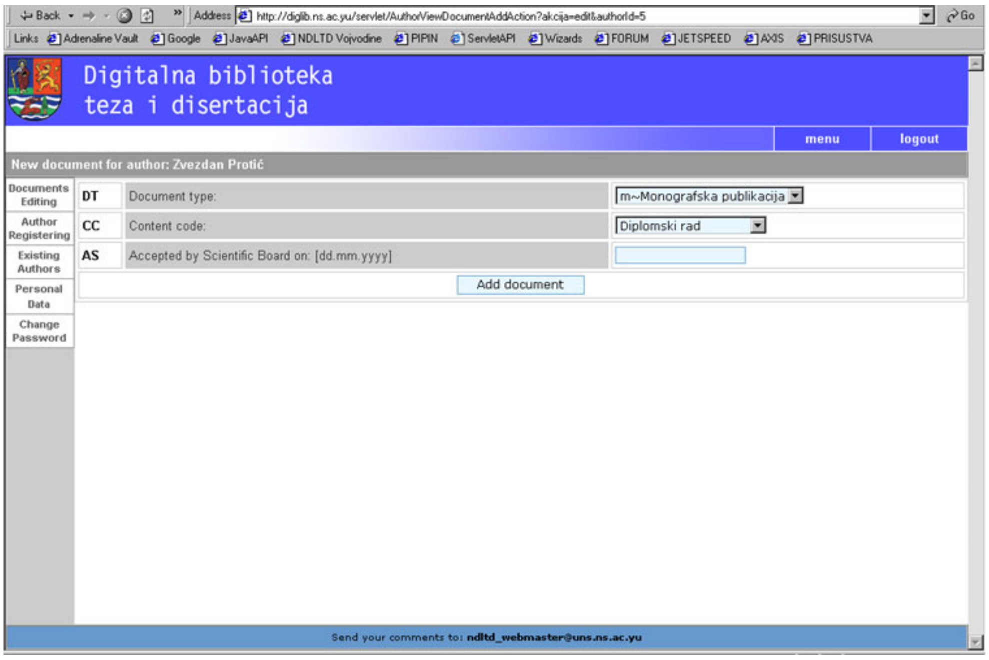
Document editing. After the document is created, it can be edited by the author. The document editing process is carried out taking the following steps:

- editing language independent metadata (Figure 4). This step consists of the following substeps:
  - editing document type
  - editing paper content

  - editing mentor metadata
  - editing language of text
  - editing year of document
  - editing physical description
  - editing scientific field
  - editing scientific discipline
  - editing universal decimal classification
  - editing date of acceptance by scientific board
- editing abstracts, extended abstracts, keywords, and annotations in the document primary language (Figure 5)
- editing abstracts, extended abstracts, keywords, and annotations in the document secondary language
- editing defended board members data (Figure 6). This step is divided into two substeps:
  - deleting existing members
  - adding new members
- uploading an electronic version of the document (Figure 7)
- editing abstracts, extended abstracts, keywords, and annotations in extended languages
- submitting the document for publishing.

These steps need not be taken all at once by the author. i.e. different steps can be taken in different phases of making of the thesis or dissertation.

Document publishing. After the author has finished editing the document, the librarian is obliged to check the document for faults, and correct them, if any. The librarian follows the same steps the author makes in the process of checking the document. When the librarian is convinced that all the metadata is correct, he publishes the document. No further corrections to the document are possible after publishing, while the document metadata and the electronic version are accessible for viewing through the FrontOffice.



http://diglib.ns.ac.yu/servlet/AuthorViewDocumentAddAction?akcija=edit&authorId=5

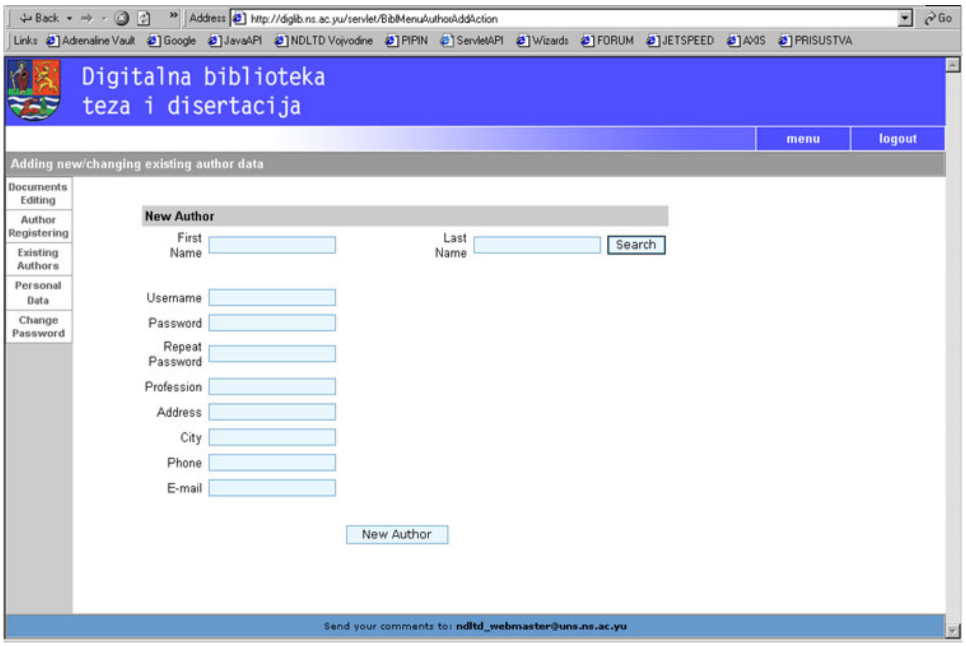
Digitalna biblioteka
teza i disertacija

New document for author: Zvezdan Protić

DT	Document type:	m~Monografska publikacija
CC	Content code:	Diplomski rad
AS	Accepted by Scientific Board on: [dd.mm.yyyy]	

Send your comments to: ndltd_webmaster@uns.ns.ac.yu

Figure 2. Creating a new document for an author



http://diglib.ns.ac.yu/servlet/BibMenuAuthorAddAction

Digitalna biblioteka
teza i disertacija

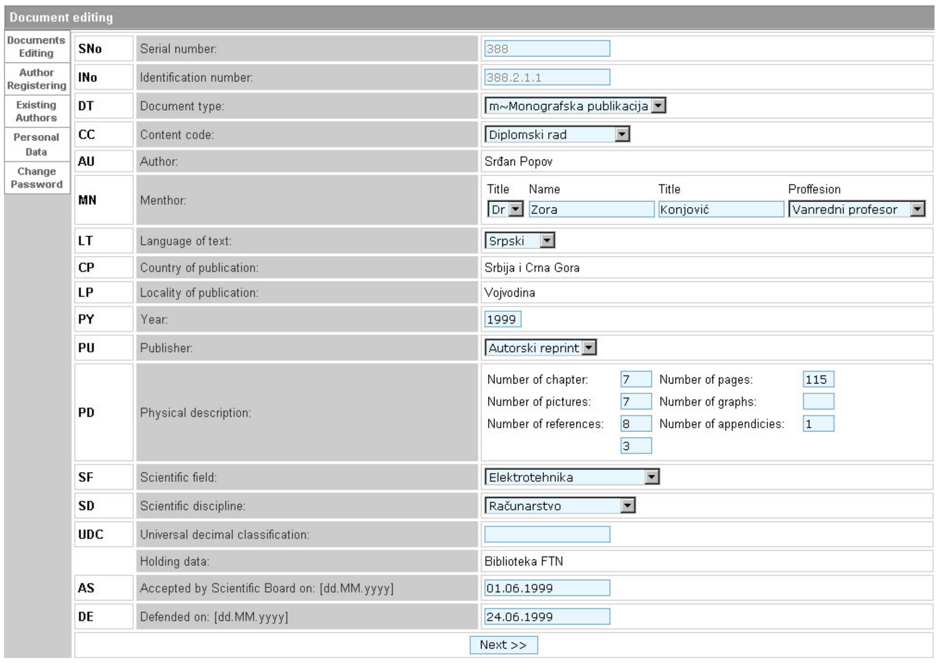
Adding new/changing existing author data

New Author

First Name
Last Name
Username
Password
Repeat Password
Profession
Address
City
Phone
E-mail

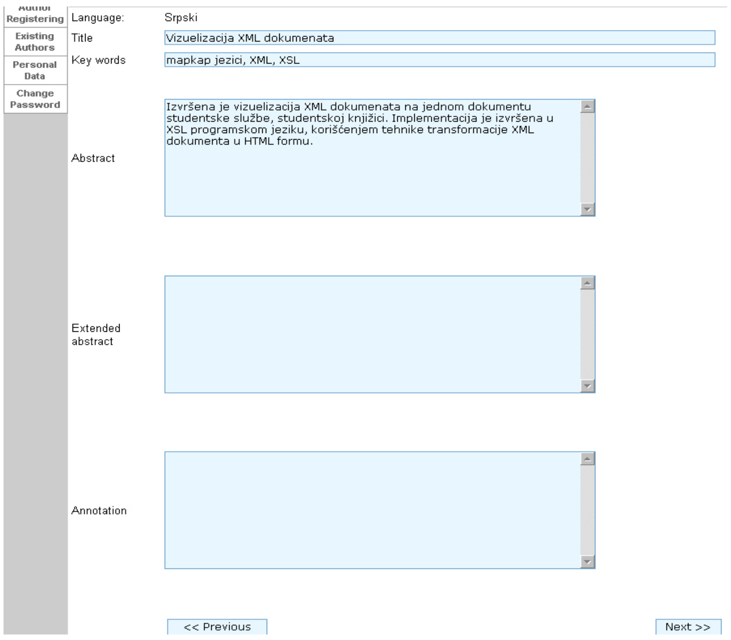
Send your comments to: ndltd_webmaster@uns.ns.ac.yu

Figure 3. Registering a new author



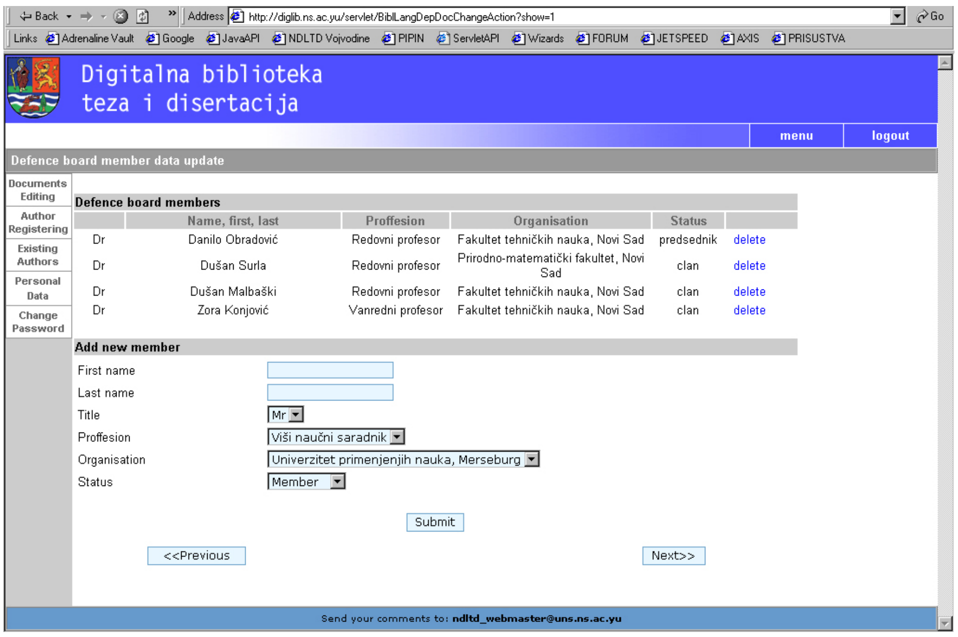
SNo	Serial number:	388
INo	Identification number:	388.2.1.1
DT	Document type:	m~Monografska publikacija
CC	Content code:	Diplomski rad
AU	Author:	Srđan Popov
MN	Menthor:	Title: Dr; Name: Zora; Title: Konjović; Proffesion: Vanredni profesor
LT	Language of text:	Srpski
CP	Country of publication:	Srbija i Crna Gora
LP	Locality of publication:	Vojvodina
PY	Year:	1999
PU	Publisher:	Autorski reprint
PD	Physical description:	Number of chapter: 7; Number of pages: 115; Number of pictures: 7; Number of graphs: ; Number of references: 8; Number of appendicies: 1; 3
SF	Scientific field:	Elektrotehnika
SD	Scientific discipline:	Računarstvo
UDC	Universal decimal classification:	
	Holding data:	Biblioteka FTN
AS	Accepted by Scientific Board on: [dd.MM.yyyy]	01.06.1999
DE	Defended on: [dd.MM.yyyy]	24.06.1999

Figure 4. Editing language independent data



Language:	Srpski
Title	Vizuelizacija XML dokumenata
Key words	mapkap jezici, XML, XSL
Abstract	Izvršena je vizuelizacija XML dokumenata na jednom dokumentu studentske službe, studentskoj knjižici. Implementacija je izvršena u XSL programskom jeziku, korišćenjem tehnike transformacije XML dokumenta u HTML formu.
Extended abstract	
Annotation	

Figure 5. Editing abstract, extended abstract, keywords and annotations


Address http://diglib.ns.ac.yu/servlet/BiblLangDepDocChangeAction?show=1

Links Adrenaline Vault Google JavaAPI NDLTD Vojvodine PIPIN ServletAPI Wizards FORUM JETSPEED AXIS PRISUSTVA

Digitalna biblioteka
teza i disertacija

menu | logout

Defence board member data update

Documents Editing | Author Registering | Existing Authors | Personal Data | Change Password

Defence board members

	Name, first, last	Proffesion	Organisation	Status	
Dr	Danilo Obradović	Redovni profesor	Fakultet tehničkih nauka, Novi Sad	predsednik	delete
Dr	Dušan Surla	Redovni profesor	Prirodno-matematički fakultet, Novi Sad	clan	delete
Dr	Dušan Malbaški	Redovni profesor	Fakultet tehničkih nauka, Novi Sad	clan	delete
Dr	Zora Konjović	Vanredni profesor	Fakultet tehničkih nauka, Novi Sad	clan	delete

Add new member

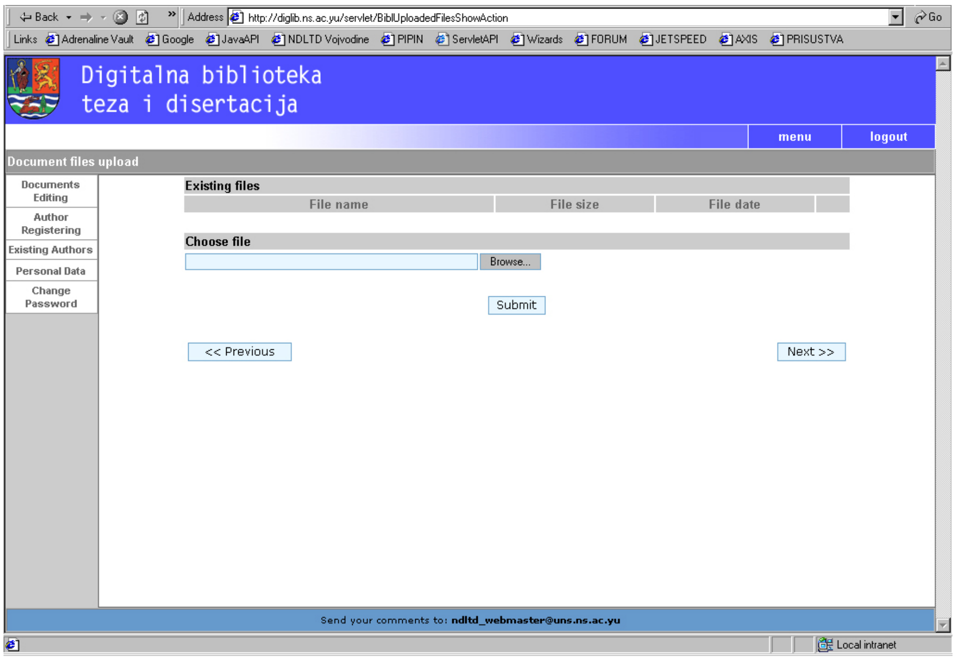
First name	
Last name	
Title	Mr
Proffesion	Viši naučni saradnik
Organisation	Univerzitet primenjenjih nauka, Merseburg
Status	Member

Submit

<<Previous Next>>

Send your comments to: **ndltd_webmaster@uns.ns.ac.yu**

Figure 6. Editing defending board members data


Address http://diglib.ns.ac.yu/servlet/BiblUploadedFilesShowAction

Links Adrenaline Vault Google JavaAPI NDLTD Vojvodine PIPIN ServletAPI Wizards FORUM JETSPEED AXIS PRISUSTVA

Digitalna biblioteka
teza i disertacija

menu | logout

Document files upload

Documents Editing | Author Registering | Existing Authors | Personal Data | Change Password

Existing files

File name	File size	File date

Choose file

Browse...

Submit

<< Previous Next >>

Send your comments to: **ndltd_webmaster@uns.ns.ac.yu**

Local intranet

Figure 7. Page for uploading an electronic version of the document

4. Conclusion

The Digital Library of Theses and Dissertations allows for quick and easy creation of documents, and editing documents metadata. Authors can register and process their theses. Librarians can register authors, create documents for authors, check documents for correctness and publish documents. Published documents are available for search at www.diglib.ns.ac.yu. The existance of language dependent and independent metadata document parts facilitates an easy system customisation for every language supported by the UTF standard.

References

[1] *Networked Digital Library of Theses and Dissertations of the University of Novi Sad*, Department of Mathematics and Informatics of the Faculty of Science, Department of Computing and Control of the Faculty of Technical Sciences, Project financed by the Provincial Secretariat for Science and Techonological Development of Autonomous Province of Vojvodina, Novi Sad, 2003.

[2] Networked Digital Library of Theses and Dissertations – NDLTD. http://www.ndltd.org

[3] Zarić, M., *User Search in the Digital Library of Theses nad Dissertations* , International Conference on Distributed Library Information Systems, Ohrid, June 1-6, 2004.

User Search in Digital Library of Theses and Dissertations of University of Novi Sad

Miroslav Zarić, Branko Milosavljević
Faculty of Technical Sciences, University of Novi Sad
{miroslavzaric,mbranko}@uns.ns.ac.yu

Abstract. This paper presents user-end application of Digital Library of Theses and Dissertations of the University of Novi Sad. The short overview of Digital Library purpose and intentions is given, accompanied with detailed explanation of User Search application functions and features. An example is given for each characteristic type of search that this application offers to end-user. Explanation of specific terms such as prefixes and document's content code is given in appendices at the end of the paper.

1. Introduction

The primary goal of the Digital Library of Theses and Dissertations of the University of Novi Sad [1,2,3] is to enable faster propagation of scientific papers produced at the University, and to simplify access to the content of graduation and master theses and dissertations to students and all other interested parties. In order to enable users to search the corresponding archive, the subsystem for user search has been developed. This subsystem is implemented as a web application, thus allowing access to the content of the Digital Library to the general public. Digital Library of Theses and Dissertations of University of Novi Sad has become part of the Networked Digital Library of Theses and Dissertations – NDLTD [4] an international organization dedicated to promoting the adoption, creation, use, dissemination and preservation of electronic analogues to the traditional paper-based theses and dissertations. Users are offered several types of search: search by keywords, by author, by content of the abstract, by content of the extended abstract, by title of the paper, by scientific field/discipline, as well as the advanced search which enables the user to create his/her own search criteria combining several different search terms. The application is multilingual, allowing easy adaption of the user interface to any language. The starting page of the application is shown in Figure 1.

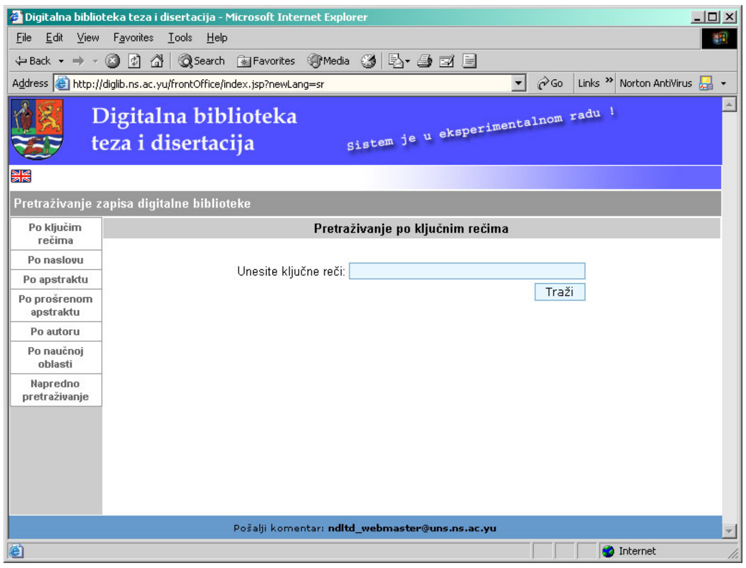

Digitalna biblioteka teza i disertacija - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://diglib.ns.ac.yu/frontOffice/index.jsp?newLang=sr
Go Links Norton AntiVirus
Digitalna biblioteka teza i disertacija
Sistem je u eksperimentalnom radu !
Pretraživanje zapisa digitalne biblioteke
Po ključim rečima
Po naslovu
Po apstraktu
Po proširenom apstraktu
Po autoru
Po naučnoj oblasti
Napredno pretraživanje
Pretraživanje po ključnim rečima
Unesite ključne reči:
Traži
Pošalji komentar: ndltd_webmaster@uns.ns.ac.yu
Internet

Figure 1. Starting web page of user search subsystem (selected language for user interface: Serbian)

As it can be seen in Figure 1, design of the web page is intentionally simplified to ensure that the user interface is simple, intuitive and easy to use. The user may, by following a link represented by the appropriate flag, change the language of the application. The main menu of the application is always available throughout the application. Following the links presented in the main menu, the user can access the pages offering different types of search:

- link *By Keywords* – enables users to enter words that should appear as keywords in resulting documents;
- link *By Title* – enables users to search for documents by the content of their titles;
- link *By Abstract* – enables users to search through the documents with search criteria based on the content of their respective abstracts;
- link *By Extended Abstract* – enables users to search through the documents by the content of their respective extended abstracts;
- link *By Author* – enables users to find documents created by specific authors;
- link *By Scientific Field* – enables users to search for documents by the scientific field of the content of documents;
- link *Advanced Search* – enables users to create their own search criteria based on the appropriate combination of search terms.

All these search types, with the exception of *Advanced Search*, represent search by the predefined criteria (prefix), see Appendix A. Prefix names and description have been largely adopted from the BISIS Library Information System [5], as well as main features of the User Search application. While searching by scientific field, user is not allowed to enter free search term, but is presented with values from the database catalogue instead.

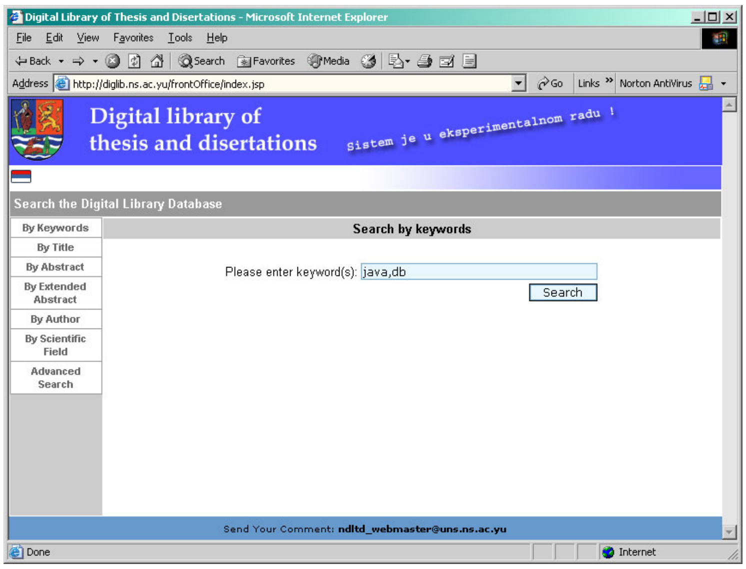
2. Search by Predefined Criteria

The following things are common for all predefined search types allowing users to enter free text:

- user is presented with one text field to enter his/her search term(s),
- system does not make a distinction between upper and lower case letters (case-insensitive search),
- punctuation marks and blanks are removed from the search term,
- use of the following wildcard characters is allowed:
- * or *%* - replaces several characters,
- *?* or _ – replaces one character,
- order of the words in the search term does not affect the result,
- documents containing all the words entered in the search term will be presented as valid results (implicit AND operation on all the words).

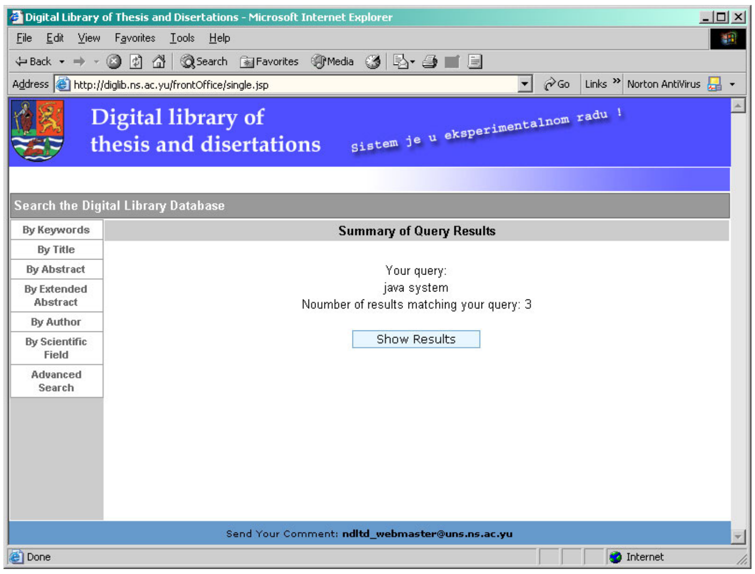
By visiting the home page of the Digital Library or by selecting the link *By keywords* the web page shown in Figure 2 is displayed. On the page there is a text field that can be filled in by the user. After the user has entered word(s) representing his/her search term, clicking the button *Search* initiates the database query searching for specified words in the appropriate metadata element. In this case the content of the prefix *KW* is queried. The content of this prefix represents words specified as keywords of the document during the process of storing the document into the Digital Library. The resulting page is shown in Figure 3. On this page the user is presented with summary of query results, displaying the submitted query and the number of documents resulting from that query. If no document is found, the user is presented with the failure notice.

By clicking the button *Show Results* the user can access the page containg the basic information about documents selected as results of the query (Figure 4). On the single page, a maximum of ten document overviews is displayed. If the number of results exceeds ten, the user can access further documents by clicking the button << *Previous* – to display the previous ten documents, or *Next* >> – to display the next ten documents.


Digital Library of Thesis and Disertations - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://diglib.ns.ac.yu/frontOffice/index.jsp
Go Links Norton AntiVirus
Digital library of thesis and disertations
sistem je u eksperimentalnom radu !
Search the Digital Library Database
By Keywords
By Title
By Abstract
By Extended Abstract
By Author
By Scientific Field
Advanced Search
Search by keywords
Search
Send Your Comment: ndltd_webmaster@uns.ns.ac.yu
Done
Internet

Please enter keyword(s): java,db

Figure 2. Web page for search by keywords
(English selected as language for user interface,
and two words entered as a search term)


Digital Library of Thesis and Disertations - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://diglib.ns.ac.yu/frontOffice/single.jsp
Go Links Norton AntiVirus
Digital library of thesis and disertations
sistem je u eksperimentalnom radu !
Search the Digital Library Database
By Keywords
By Title
By Abstract
By Extended Abstract
By Author
By Scientific Field
Advanced Search
Summary of Query Results
Show Results
Send Your Comment: ndltd_webmaster@uns.ns.ac.yu
Done
Internet

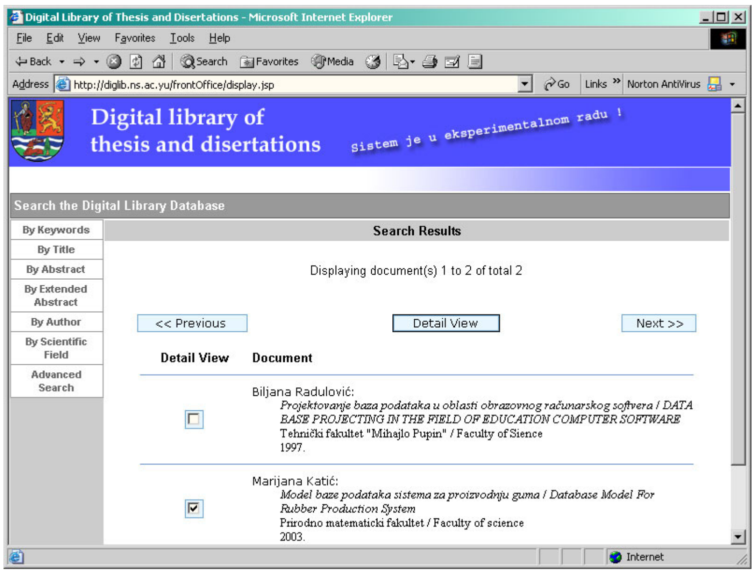
Your query:
java system
Noumber of results matching your query: 3

Figure 3. Web page presenting the summary of query results

Information about the document author, title (in every language supplied), storage place and year of publication is displayed for each document. Each document is accompanied by a check box that enables the user to select a document for detailed view. With the help of this overview page, the user can easily evaluate relevance of every document to his/her query, and decide on the document (s)he wants to see in details, or make a new query altogether. After the user has selected appropriate documents, clicking the button *Detail View* takes him/her to the page with detailed information about selected documents. The web page with document details is shown in Figure 5.

In the detailed view, beside already mentioned data, content code (see Appendix B), document abstract and extended abstract are also presented. The most important data on this page is a hyperlink to the electronic version of the original document. The electronic version of the original document is usually in *Adobe pdf* format, although other formats can be used as well. As in the overview of the documents, one page presents data of up to ten documents, and if there are more results, buttons << *Previous* or *Next* >> are used to traverse through the rest of the documents.

The user is allowed, at any given moment in time, from any page, to start a new query by selecting the appropriate link from the main menu. Search by title, author, abstract, extended abstract follows the same pattern as the search by keywords, only the prefix the database is queried on is changed.


Digital Library of Thesis and Disertations - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://diglib.ns.ac.yu/frontOffice/display.jsp
Go Links Norton AntiVirus
Digital library of thesis and disertations
Sistem je u eksperimentalnom radu !
Search the Digital Library Database
By Keywords
By Title
By Abstract
By Extended Abstract
By Author
By Scientific Field
Advanced Search
<< Previous
Detail View
Next >>
Internet

Search Results

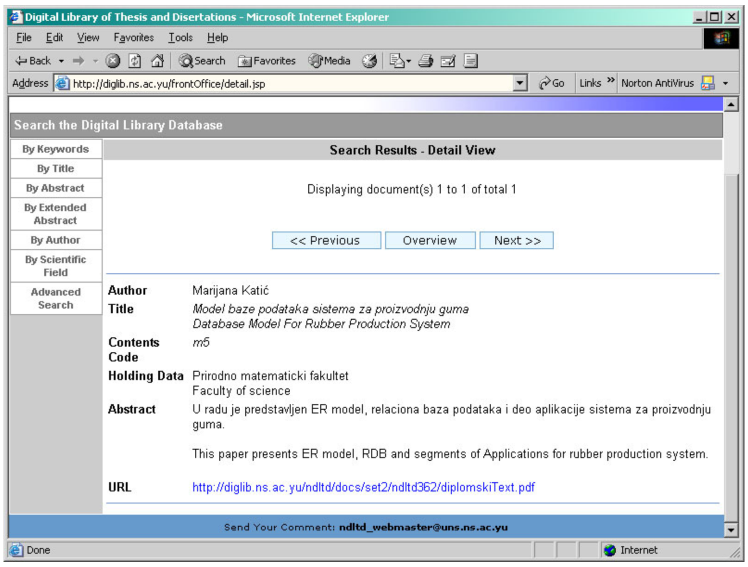
Displaying document(s) 1 to 2 of total 2

Detail View	Document
☐	Biljana Radulović: *Projektovanje baza podataka u oblasti obrazovnog računarskog softvera / DATA BASE PROJECTING IN THE FIELD OF EDUCATION COMPUTER SOFTWARE* Tehnički fakultet "Mihajlo Pupin" / Faculty of Sience 1997.
☑	Marijana Katić: *Model baze podataka sistema za proizvodnju guma / Database Model For Rubber Production System* Prirodno matematicki fakultet / Faculty of science 2003.

Figure 4. Web page displaying overview of the document found

3. Search by Scientific Field

Strictly speaking this type of search is actually the combined search by two search criteria: SF (scientific field) and SD (scientific discipline). In contrast to the previously discussed search methods, in this case the user cannot enter the search term itself, but is instead presented with values from the controlled vocabulary stored in the database of the Digital Library. Values displayed in drop-down menus are context sensitive, i.e. changing the selection in the scientific field menu loads appropriate disciplines in the lower menu. Figure 6 displays the web page for search by scientific field. The process of selecting the scientific discipline for the selected scientific field is displayed in Figure 7. After the user has selected the appropriate scientific field and discipline, clicking the button *Search* initiates the database query.


Digital Library of Thesis and Disertations - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://diglib.ns.ac.yu/frontOffice/detail.jsp
Go Links Norton AntiVirus
By Keywords
By Title
By Abstract
By Extended Abstract
By Author
By Scientific Field
Advanced Search
<< Previous
Overview
Next >>
Done
Internet

Search the Digital Library Database

Search Results - Detail View

Displaying document(s) 1 to 1 of total 1

Author	Marijana Katić
Title	*Model baze podataka sistema za proizvodnju guma* *Database Model For Rubber Production System*
Contents Code	*m5*
Holding Data	Prirodno matematicki fakultet Faculty of science
Abstract	U radu je predstavljen ER model, relaciona baza podataka i deo aplikacije sistema za proizvodnju guma. This paper presents ER model, RDB and segments of Applications for rubber production system.
URL	http://diglib.ns.ac.yu/ndltd/docs/set2/ndltd362/diplomskiText.pdf

Send Your Comment: **ndltd_webmaster@uns.ns.ac.yu**

Figure 5. Web page displaying details about selected document(s)

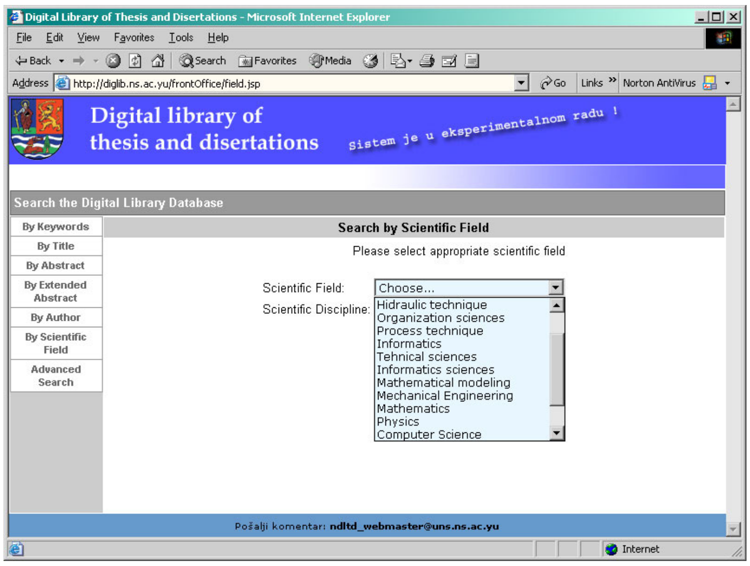

Digital Library of Thesis and Disertations - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://diglib.ns.ac.yu/frontOffice/field.jsp
Digital library of thesis and disertations
Sistem je u eksperimentalnom radu !
Search the Digital Library Database
By Keywords
By Title
By Abstract
By Extended Abstract
By Author
By Scientific Field
Advanced Search
Search by Scientific Field
Please select appropriate scientific field
Scientific Field: Choose...
Scientific Discipline:
Hidraulic technique
Organization sciences
Process technique
Informatics
Tehnical sciences
Informatics sciences
Mathematical modeling
Mechanical Engineering
Mathematics
Physics
Computer Science
Pošalji komentar: ndltd_webmaster@uns.ns.ac.yu
Internet

Figure 6. Selecting scientific field

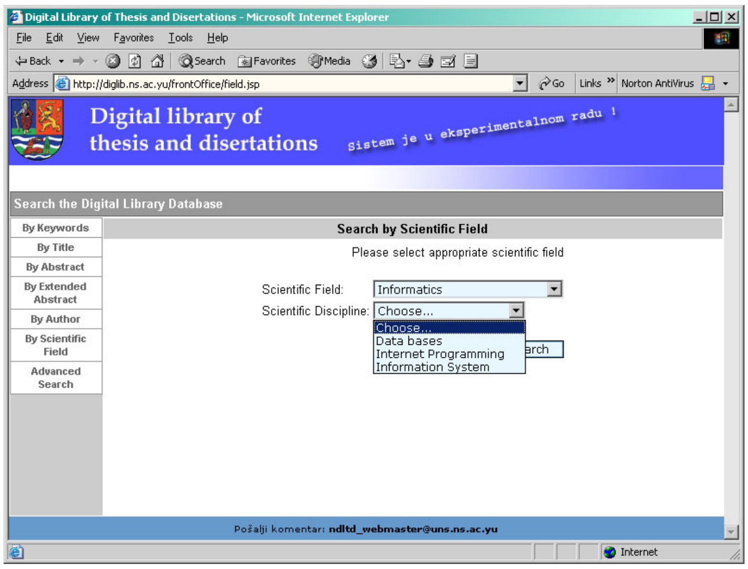
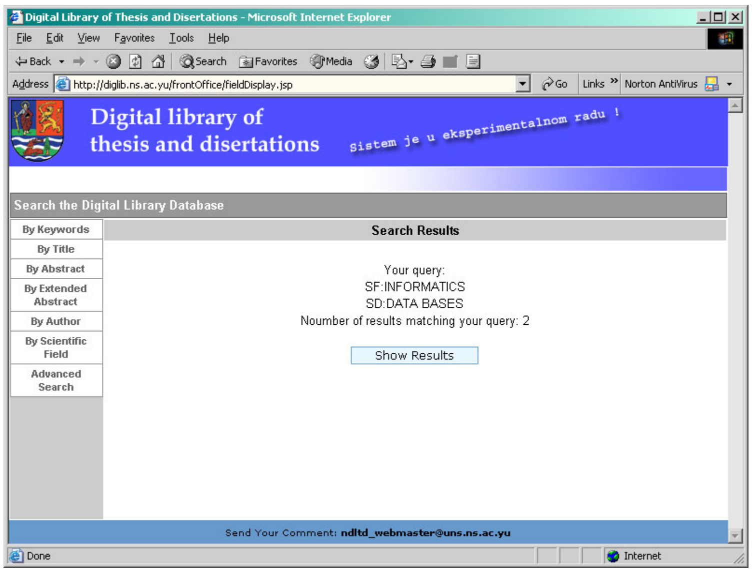

Digital Library of Thesis and Disertations - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://diglib.ns.ac.yu/frontOffice/field.jsp
Digital library of thesis and disertations
Sistem je u eksperimentalnom radu !
Search the Digital Library Database
By Keywords
By Title
By Abstract
By Extended Abstract
By Author
By Scientific Field
Advanced Search
Search by Scientific Field
Please select appropriate scientific field
Scientific Field: Informatics
Scientific Discipline: Choose...
Choose...
Data bases
Internet Programming
Information System
Pošalji komentar: ndltd_webmaster@uns.ns.ac.yu
Internet

Figure 7. Selecting scientific discipline

As a result, only the document whose content is categorised as belonging to the specified scientific field/discipline is selected. This information is supplied by the author during the placement of the document into the Digital Library. After the database query is done, the web page presenting the summary of query results is displayed (Figure 8). Further steps to view the document overview and details are analogue to the search by keyword. The provincial government has initiated a process of standardising the categorisation of scientific fields and disciplines, to adapt internal standards to the EU standards.



Digital Library of Thesis and Disertations - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://diglib.ns.ac.yu/frontOffice/fieldDisplay.jsp
Go Links Norton AntiVirus
Digital library of thesis and disertations
sistem je u eksperimentalnom radu !
Search the Digital Library Database
By Keywords
By Title
By Abstract
By Extended Abstract
By Author
By Scientific Field
Advanced Search
Search Results
Show Results
Send Your Comment: ndltd_webmaster@uns.ns.ac.yu
Done
Internet

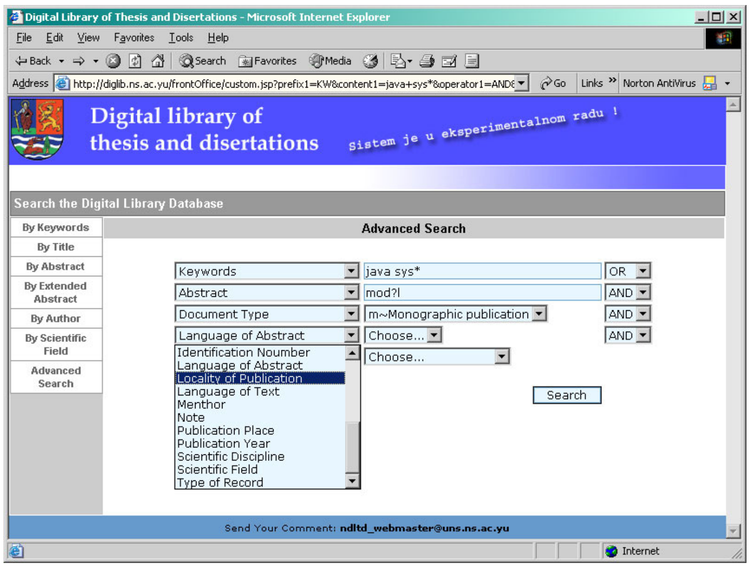
Your query:
SF:INFORMATICS
SD:DATA BASES
Noumber of results matching your query: 2

Figure 8. Summary of query results returned after the search on scientific field/discipline

4. Advanced Search

After selecting the link *Advanced Search* from the main menu, the web page shown in Figure 9 is displayed. On this page the user can create his/her own search criteria by combining search on specific fields.

The maximum of five separate search terms can be combined. The selection of search criteria against which the corresponding text field will be matched is made by drop-down menus. In these drop-down menus, all fields by which the search can be done are listed. For some fields the user is allowed to enter free search terms, while for the other user is presented (in the form of a drop-down menu) with values retrieved from the database catalogue. Linking the content of one field with the rest of the combined query is done by using the logical operators (AND, OR, NOT). The operator AND links two fields in such way that resulting documents must contain values of both these fields. The operator OR links two fields so that resulting documents must contain values of at least one of these fields, while operator NOT links two fields in such way that resulting documents must contain the value of the first field, but must not contain values stated in the second one. If any field is left empty, it is ignored.


Digital Library of Thesis and Disertations - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Go Links Norton AntiVirus
Internet

Address http://diglib.ns.ac.yu/frontOffice/custom.jsp?prefix1=KW&content1=java+sys*&operator1=AND

Digital library of thesis and disertations

Sistem je u eksperimentalnom radu !

Search the Digital Library Database

By Keywords
By Title
By Abstract
By Extended Abstract
By Author
By Scientific Field
Advanced Search

Advanced Search

Keywords	java sys*	OR
Abstract	mod?l	AND
Document Type	m~Monographic publication	AND
Language of Abstract	Choose...	AND
Locality of Publication	Choose...	

Identification Noumber
Language of Abstract
Locality of Publication
Language of Text
Menthor
Note
Publication Place
Publication Year
Scientific Discipline
Scientific Field
Type of Record

Search

Send Your Comment: ndltd_webmaster@uns.ns.ac.yu

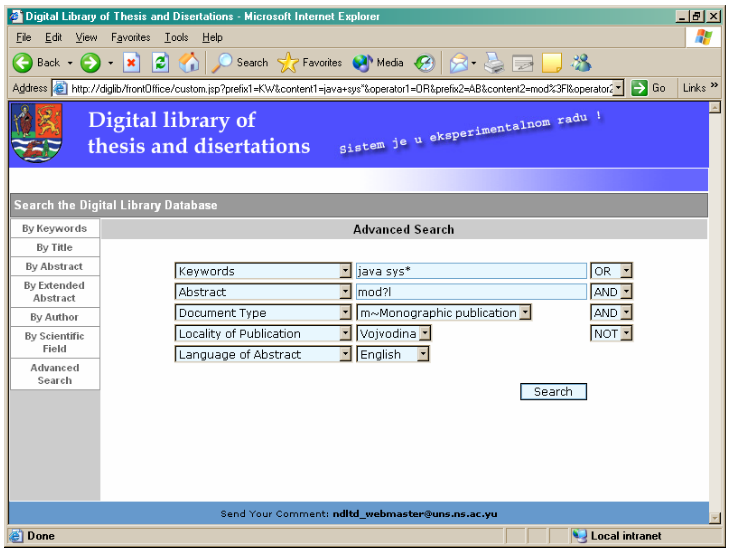
Figure 9. Creating a combined query

The example shown in Figure 10 shows a combined query which is used to find document(s) respecting the following restrictions:

- Keywords of the document must contain the word *java* and words begining with *sys* **OR**
- Abstract of the document must contain words that are similar to *mod?l* (for example – model, modul) **AND**
- Document type is a monographic publication **AND**
- Locality of publication must be *Vojvodina* and
- Original language of abstract **IS NOT** english.

As it can be seen from the example, values for the document type, locality of publication and language of abstract are retrieved from the database catalogue, while terms for keywords and abstract are left for the user to enter.

When the user has prepared his/her combined query, the search through the database is initiated by clicking the button *Search*. As a result, the web page with the summary of query results is displayed (Figure 11).


Digital Library of Thesis and Disertations - Microsoft Internet Explorer

Address: http://diglib/frontOffice/custom.jsp?prefix1=KW&content1=java+sys*&operator1=OR&prefix2=AB&content2=mod%3Fl&operator2

Digital library of thesis and disertations

sistem je u eksperimentalnom radu !

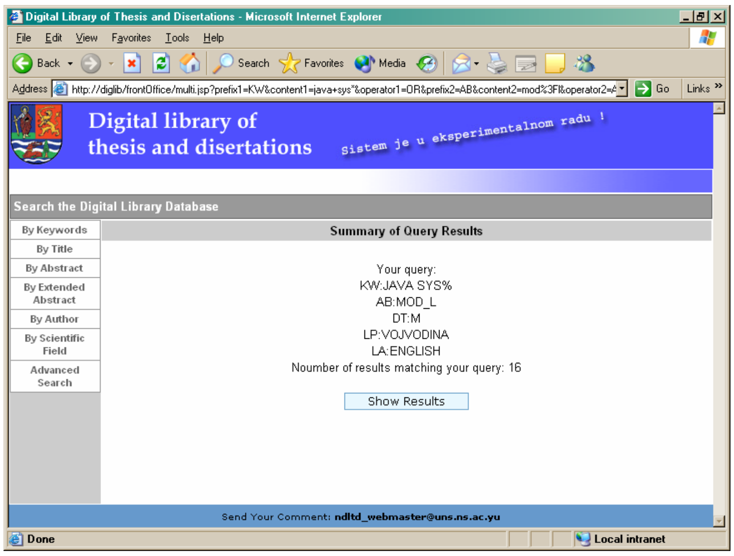
Search the Digital Library Database

Advanced Search

Keywords	java sys*	OR
Abstract	mod?l	AND
Document Type	m~Monographic publication	AND
Locality of Publication	Vojvodina	NOT
Language of Abstract	English	

Send Your Comment: ndltd_webmaster@uns.ns.ac.yu

Figure 10. Prepared combined query for advanced search


Digital Library of Thesis and Disertations - Microsoft Internet Explorer

Address: http://diglib/frontOffice/multi.jsp?prefix1=KW&content1=java+sys*&operator1=OR&prefix2=AB&content2=mod%3Fl&operator2=A

Digital library of thesis and disertations

sistem je u eksperimentalnom radu !

Search the Digital Library Database

Summary of Query Results

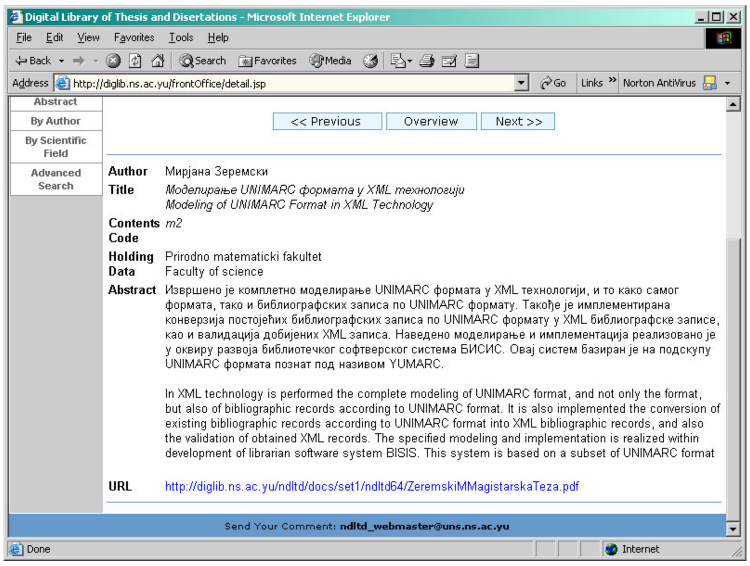
Your query:
KW:JAVA SYS%
AB:MOD_L
DT:M
LP:VOJVODINA
LA:ENGLISH
Noumber of results matching your query: 16

Send Your Comment: ndltd_webmaster@uns.ns.ac.yu

Figure 11. Summary of query results

The further steps of viewing the document overview and details are analogue to previous search methods.

During the development of the Digital Library, much attention is paid to the problem of creating a functional multilingual application. The application allows for easy adoption of any language for user interface, with no code changes necessary. All that needs to be done is to create a specific file with translation of the interface messages to the desired language. The system is also fully capable of storing metadata in different languages, due to use of the Unicode character set in the database tables. Figure 12 shows an example of a document whose metadata are entered both in latin and cyrillic alphabet.


Digital Library of Thesis and Disertations - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://diglib.ns.ac.yu/frontOffice/detail.jsp
Go Links Norton AntiVirus
Abstract
By Author
By Scientific Field
Advanced Search
<< Previous
Overview
Next >>
Done
Internet

Author Мирјана Земерски

Title *Моделирање UNIMARC формата у XML технологији*
Modeling of UNIMARC Format in XML Technology

Contents Code *m2*

Holding Data Prirodno matematicki fakultet
Faculty of science

Abstract Извршено је комплетно моделирање UNIMARC формата у XML технологији, и то како самог формата, тако и библиографских записа по UNIMARC формату. Такође је имплементирана конверзија постојећих библиографских записа по UNIMARC формату у XML библиографске записе, као и валидација добијених XML записа. Наведено моделирање и имплементација реализовано је у оквиру развоја библиотечког софтверског система БИСИС. Овај систем базиран је на подскупу UNIMARC формата познат под називом YUMARC.

In XML technology is performed the complete modeling of UNIMARC format, and not only the format, but also of bibliographic records according to UNIMARC format. It is also implemented the conversion of existing bibliographic records according to UNIMARC format into XML bibliographic records, and also the validation of obtained XML records. The specified modeling and implementation is realized within development of librarian software system BISIS. This system is based on a subset of UNIMARC format

URL http://diglib.ns.ac.yu/ndltd/docs/set1/ndltd64/ZeremskiMMagistarskaTeza.pdf

Send Your Comment: **ndltd_webmaster@uns.ns.ac.yu**

Figure 12. Usage of Unicode enables the system to equally handle different scripts

5. Conclusion

Searching the Digital Library of Theses and Dissertations at the University of Novi Sad is performed via the system's web site. Searching can be performed by a predefined citeria, such as document title or abstract, or by an advanced criteria comprising multiple search terms on multiple prefixes. Hence, the web site provides the means for searching the bibliographic database in different ways, thus enabling users to choose a searching mode best suited for their needs and the level of skill.

The web site fully supports internationalisation. All textual messages are displayed in a user-chosen language. Each visitor of the web site can choose its own language for the user interface. Besides, all text that is displayed and entered is stored and handled using Unicode character set. This way the system is capable of handling queries and text content written in different scripts.

References

[1] *Digital Library of Theses and Dissertations of the University of Novi Sad*. http://diglib.ns.ac.yu

[2] *Networked Digital Library of Theses and Dissertations*. Dušan Surla, ed., Executive Council of Autonomous Province of Vojvodina, Provincial Secretariat for Science and Techonological Development, Novi Sad, 2003.

[3] D. Surla, Z. Konjović, B. Milosavljević, G. Sladić, Z. Protić, S. Komazec, D. Okanović. *An Overview of the Implementation of the Networked Digital Library of Theses and Dissertations*. Infoteka 5(1), 2004.

[4] *Networked Digital Library of Theses and Dissertations – NDLTD*. http://www.ndltd.org

[5] D. Surla, Z. Konjović, eds. *Distributed Library Information System BISIS*. Grupa za informacione tehnologije, Novi Sad, 2004. (in Serbian)

Appendix A. List of Search Prefixes

Prefix	**Description**
KW	Keywords
TI	Title
AB	Abstract
AX	Extended Abstract
AU	Author
AN	Accesion Number
AS	Accepted by Scientific Board
CC	Contents Code
CP	Country of Publication
DB	Thesis Defend Board
DE	Defended
DT	Document Type
HD	Holding Data
IN	Identification Number
LA	Language of Abstract
LP	Locality of Publication
LT	Language of Text
MN	Mentor
NO	Note

PP	Publication Place
PY	Publication Year
SD	Scientific Discipline
SF	Scientific Field
TR	Type of Record

Appendix B. Content Codes

Code	Description
m	PhD Dissertation
m2	MSc Thesis
m5	Graduation Thesis

Metadata Dissemination Using OAI-PMH

Miroslav Zarić[1], Dušan Surla[2]
[1]Faculty of Technical Sciences, University of Novi Sad
[2]Faculty of Science and Mathematics, University of Novi Sad
{miroslavzaric,surla}@uns.ns.ac.yu

Abstract. The paper presents partial results of the project of the Networked Digital Library of Theses, Dissertations and Graduate Theses of the University of Novi Sad, specifically the implementation of the Protocol for Meatadata Harvesting (PMH) for metadata dissemination in *Dublin Core* format, as well as short overview of the Open Archives Initiative (OAI) and its Protocol for Metadata Harvesting. Short analysis of possible usage of PMH in libraries in our country and the reference to a list of NDLTD member libraries are also given. These libraries are using PMH to support services for metadata gathering.

1. Introduction

The paper presents partial results of the project of the Networked Digital Library of Theses, Dissertations and Graduate Theses of the University of Novi Sad [1]. The project has been financed by the Provincial Secretariat for Science and Technological Development of the Autonomous Province of Vojvodina. It has been assigned to the University of Novi Sad, which has joined the *Networked Digital Library of Theses and Dissertations - NDLTD* [2] during 2003. This initiative is based on the *Open Archive Initiative – OAI* [3]. The primary goal of the OAI is to enable efficient search and interchange of documents stored in different archives on the Internet. In order to achieve this goal within an acceptable time limit, the significant effort is being made to the standardise metadata formats, and to specify the interchange protocol which needs to be relatively easy to implement. Within the initiative, the consensus has been reached that all memebers must expose their metadata in the non-qualified *Dublin Core* format, and the *Protocol for Metadata Harvesting* [4] has been developed and adopted as the standard for metadata interchange.

The main topic of this paper is the implementation of PMH, in its part defining protocol communication features and metadata representation in the *Dublin Core* format. The minimal technical requirements for data providers are disscused. Short disscusion of the implementation guidelines for client – side application developers is also given.

1.1. Short Overview of the OA Initiative and of PMH features

The Open Archives Initiative develops and promotes interoperability standards which aim at facilitating efficient dissemination of the content. The Open Archives Initiative has its roots in an effort to enhance access to e-print archives as a means of increasing the availability of scholarly papers. The first official result of the OA Initiative was adoption of the *Santa Fe Convention* [5] at the meeting held in 1999, attended by participants representing organisations already using, or organisations planning to use e-print archives, as publicly accessible archives, as well as organisations which were interested in developing value added services for these archives (search engines, citation search etc). The convention was a simple technical and organisational framework with the main goal to support interoperability between e-print archives. All participants stated their intention to implement these guidelines in a relatively short period, thus allowing first tests of data interchange to take place during the year 2000. At the same time a public invitation to join the initiative has been issued to all interested parties. The first archive to become publicly available was arXiv.org, an archive of electronic publications of the national Institute in Los Alamos (USA), created by Paul Ginsparg, which has become a crucial hub for communicating research findings in physics. The *Santa Fe Convention* has been phased out in the mean time and the *Protocol for Metadata Harvesting* has been developed instead.

1.1.1. Basic concepts of the OA initiative

- Community of electronic publishers, where the term “archive” is used as a repository for stored information.
- Protocol for metadata harvesting defines basic framework for metadata interchange within the OA initiative
- Each member of the initiative can take part playing one of the two roles:
    - *Data providers*
    - *Service providers*
- The term “open” in the OAI should be viewed from the architectural perspective – defining and promoting machine interfaces facilitating the availability of the content to a variety of providers. Openness does not mean “free” or “unlimited” access to the information repositories conforming to the OAI-PMH.

1.1.2. Basic features of the PMH protocol

- The Open Archives Initiative Protocol for Metadata Harvesting provides an application-independent interoperability framework based on *metadata harvesting* in the form of XML documents.

- Each participant has freedom of choice regarding the hardware platform and software systems upon which the OAI-PMH application is built
- The complete specification of XML documents being interchanged is given, as well as definitions of all concepts regarding the protocol and its implementation within the OAI
- HTTP based, functions on the request-response principle:
    - PMH requests are sent embedded as parameters within HTTP GET or POST requests
    - Response must be in the text/xml format

1.1.3. Basic concepts of PMH

- A *harvester* is a client application which issues OAI-PMH requests. A harvester is operated by a service provider as a means of collecting metadata from repositories;
- A *repository* is a network accessible server which can process 6 OAI-PMH requests in the manner described in this document. A repository is managed by a data provider to expose metadata to harvesters;
- An *item* is a constituent of a repository from which metadata about a resource can be disseminated. Each item has an identifier which is unique within the scope of the repository of which it is a constituent;
- A *unique identifier* unambigiously identifies an item within a repository; the unique identifier is used in OAI-PMH requests for extracting metadata from the item;
- A *record* is metadata expressed in a single format. A record is returned in an XML-encoded byte stream in response to an OAI-PMH request for metadata from an item;
- A *set* is an optional construct for grouping items for the purpose of selective harvesting;
- *Selective harvesting* allows harvesters to limit harvest requests to portions of the metadata available from a repository.

OAI-PMH provides for metadata interchange in different formats, although the only required format is the unqualified *Dublin Core*. The protocol does not make any assumptions about the desired format, but specifies an XML element metadata should reside within.

OAI-PMH supports flow-control by implementing the mechanism of *resumption tokens*. Some requests result in so-called list response, i.e. a list of discrete entities, which in some cases can be very long. In this case only partial result (incomplete) is returned to the user, along with the *resumption token*. In order to make the response a *complete list*, the harvester will need to issue one or more requests with `resumptionTokens` as arguments. The complete list then consists of the concatenation of the *incomplete lists* from the sequence of requests, known as a *list request sequence*. This flow control mechanism, in combination with HTTP transport layer facilities, provides some basic tools with which a repository can enforce an *acceptable use policy* for its harvesting interface. *Resumption token* can be time-limited.

OAI-PMH specifies an error handling mechanism. For each request there is a defined set of required and optional parameters, and its valid values, and if the request does not conform to the specification, the client application will receive an XML document with an error element specifing the error code and the error description. In this way some basic error-correction capabilities can be built in into the client application.

There are six specified OAI-PMH requests, and their short description and returned results are given below:

- **GetRecord** – This verb is used to retrieve an individual metadata record from a repository. Required arguments specify the identifier of the item the record is requested from and the format of the metadata that should be included in the record.
- **Identify** – This verb is used to retrieve information about a repository. Repositories may also employ the Identify verb to return additional descriptive information.
- **ListIdentifiers** – This verb is an abbreviated form of ListRecords, retrieving only headers rather than records. Required parameter is metadataFormat, optional arguments permit selective harvesting of headers based on set membership and/or datestamp.
- **ListMetadataFormats** – This verb is used to retrieve the metadata formats available from a repository. An optional argument restricts the request to the formats available for a specific item.
- **ListRecords** – This verb is used to harvest records from a repository. Optional arguments permit selective harvesting of records based on set membership and/or datestamp.
- **ListSets** – This verb is used to retrieve the set structure of a repository, useful for selective harvesting.

2. Dublin Core

The *Dublin Core* is the only required metadata format each OAI data provider must implement. The *Dublin Core* specification can be found at the address given in [6]. The OAI-PMH specifies the use of so-called unspecified Dublin Core format, and the metadata element within response XML document must validate against XML schema given at:

http://www.openarchives.org/OAI/2.0/oai_dc.xsd.

Although within the *Dublin Core* project there is a specification of additional elements, the OAI-PMH specifies usage of following dc elements:

- **dc:title** – Title of the document, the name under which document is published and is generally known.
- **dc:creator** – Name of the person primarily responsible for document content.
- **dc:subject** – A topic of the content of the resource.
- **dc:description** – An account of the content of the resource (an abstract, table of contents, reference to a graphical representation of content or a free-text account of the content).
- **dc:publisher** – An entity responsible for making the resource available.
- **dc:contributor** – An entity responsible for making contributions to the content of the resource.
- **dc:date** – A date of an event in the lifecycle of the resource. Typically, Date will be associated with the creation or availability of the resource.
- **dc:type** – The nature or genre of the content of the resource.
- **dc:identifier** – An unambiguous reference to the resource within a given context.
- **dc:source** – A Reference to a resource from which the present resource is derived.
- **dc:language –** A language of the intellectual content of the resource.
- **dc:relation –** A reference to a related resource.
- **dc:coverage –** The extent or scope of the content of the resource.
- **dc:rights** – Information about rights held in and over the resource

Due to its relatively simple structure, the *Dublin Core* metadata format has been selected as the basic format for metadata interchange within the OAI. Its simplicity should allow simple record conversion from the local format, and at the same time its metadata set should be descriptive and representative enough to allow for successful harvesting of the archive. At the 3rd OAI Workshop, held at CERN, Geneva in 2004, the positive impact of the Dublin Core simplicity on the popularisation and overall acceptance of the OAI was stressed, but at the same time warnings were issued that the non-compulsory nature of the elements of its element set and their unspecified repetitiveness cast a shadow on the quality of metadata presented by different archives. This could create problems for metadata based document discovery. To cope with this problem, authors (they are usually in charge of creating metadata about their papers) are strongly advised to fill in the metadata set with as many details as possible, and development of the initiative itself could, at some point in the future, provide strict rules regarding element set completeness (some institutional repositories already have their own internal policy in place). Apart from the problem of the document discovery, questionable quality of metadata representing documents also create problems in the process of eliminating duplicates.

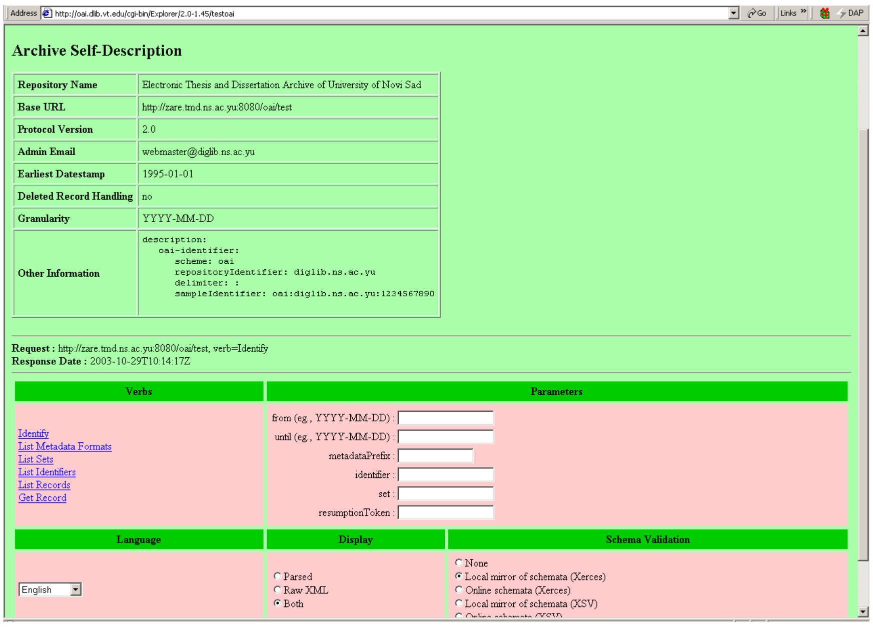
2.1. Protocol Implementation – Short Examples

The protocol has been implemented in the Java environment. Figures 1-3 display server responses to some client requests. For testing purposes the *Repository Explorer* [7] tool of the OA Initiative has been used. This tool can be used to manually issue each request, along with the required and optional parameters. The response can be displayed both formatted and unformatted (raw XML). The server which has been submitted to the test was the one on which our PMH data provider subsystem implementation was developed. The common parameter for each request is ***verb*** which identifies a specific request sent by the client application – the value of this verb represents the action that should be taken.

Figure 1 shows the server response to the ***Identify*** request. The required parameter is ***verb*** with the value ***Identify***. Other parameters are not allowed in this request. The Figure shows the basic information returned by the server describing the repository server of the Digital Library of Theses and Dissertations of University of Novi Sad.

Within the ***ListMetadataFormats*** request only the required parameter ***verb*** is sent with the value ***ListMetadataFormats.*** Figure 2 shows the parsed and the raw response returned by the server. It can be seen that our server is currently able to disseminate metadata only in the *DublinCore* format.

For the **GetRecord** request the required parameters are ***verb***, ***identifier*** and ***metadataPrefix***. As a response, the client application will receive the content of a specified record in the specified metadata format (*DublinCore)*. The Figure 3 shows one record received as a result of this request. The PMH specifies that an XML document must be UTF-8 encoded, thus allowing different languages to be displayed correctly, as shown in the Figure.



Address http://oai.dlib.vt.edu/cgi-bin/Explorer/2.0-1.45/testoai

Archive Self-Description

Repository Name	Electronic Thesis and Dissertation Archive of University of Novi Sad
Base URL	http://zare.tmd.ns.ac.yu:8080/oai/test
Protocol Version	2.0
Admin Email	webmaster@diglib.ns.ac.yu
Earliest Datestamp	1995-01-01
Deleted Record Handling	no
Granularity	YYYY-MM-DD
Other Information	description: oai-identifier: scheme: oai repositoryIdentifier: diglib.ns.ac.yu delimiter: : sampleIdentifier: oai:diglib.ns.ac.yu:1234567890

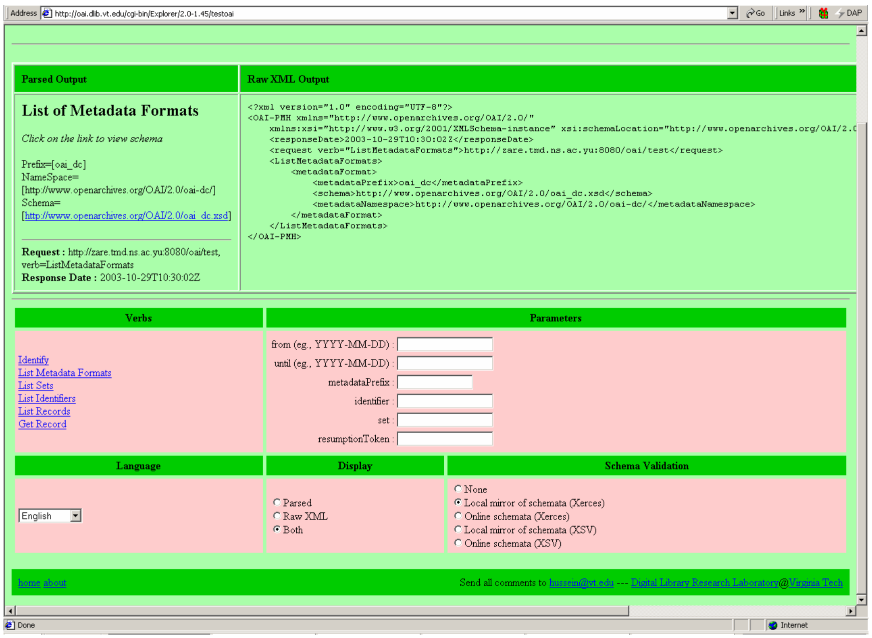
```
description:
   oai-identifier:
      scheme: oai
      repositoryIdentifier: diglib.ns.ac.yu
      delimiter: :
      sampleIdentifier: oai:diglib.ns.ac.yu:1234567890
```

Request : http://zare.tmd.ns.ac.yu:8080/oai/test, verb=Identify
Response Date : 2003-10-29T10:14:17Z

Verbs	Parameters
Identify List Metadata Formats List Sets List Identifiers List Records Get Record	from (eg., YYYY-MM-DD) : until (eg., YYYY-MM-DD) : metadataPrefix : identifier : set : resumptionToken :

Language	Display	Schema Validation
English	Parsed Raw XML [✓ Both]	None [✓ Local mirror of schemata (Xerces)] Online schemata (Xerces) Local mirror of schemata (XSV)

Figure 1. Server response to *Identify* request



Address http://oai.dlib.vt.edu/cgi-bin/Explorer/2.0-1.45/testoai

Parsed Output	Raw XML Output
List of Metadata Formats *Click on the link to view schema* Prefix=[oai_dc] NameSpace= [http://www.openarchives.org/OAI/2.0/oai-dc/] Schema= [http://www.openarchives.org/OAI/2.0/oai_dc.xsd] **Request :** http://zare.tmd.ns.ac.yu:8080/oai/test, verb=ListMetadataFormats **Response Date :** 2003-10-29T10:30:02Z	see below

```
<?xml version="1.0" encoding="UTF-8"?>
<OAI-PMH xmlns="http://www.openarchives.org/OAI/2.0/"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.openarchives.org/OAI/2.0
    <responseDate>2003-10-29T10:30:02Z</responseDate>
    <request verb="ListMetadataFormats">http://zare.tmd.ns.ac.yu:8080/oai/test</request>
    <ListMetadataFormats>
        <metadataFormat>
            <metadataPrefix>oai_dc</metadataPrefix>
            <schema>http://www.openarchives.org/OAI/2.0/oai_dc.xsd</schema>
            <metadataNamespace>http://www.openarchives.org/OAI/2.0/oai-dc/</metadataNamespace>
        </metadataFormat>
    </ListMetadataFormats>
</OAI-PMH>
```

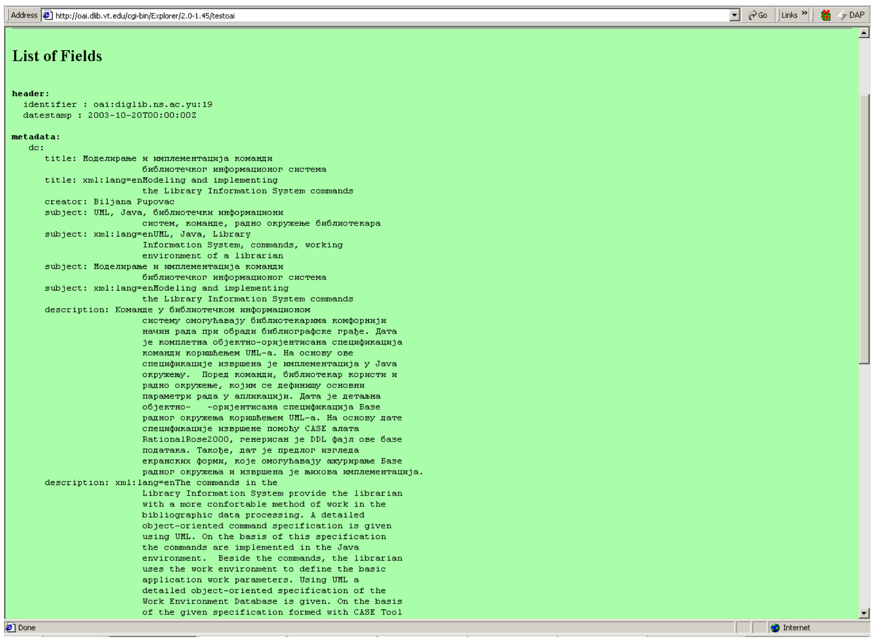
Verbs	Parameters
Identify List Metadata Formats List Sets List Identifiers List Records Get Record	from (eg., YYYY-MM-DD) : until (eg., YYYY-MM-DD) : metadataPrefix : identifier : set : resumptionToken :

Language	Display	Schema Validation
English	Parsed Raw XML [✓ Both]	None [✓ Local mirror of schemata (Xerces)] Online schemata (Xerces) Local mirror of schemata (XSV) Online schemata (XSV)

home about — Send all comments to hussein@vt.edu --- Digital Library Research Laboratory@Virginia Tech

Done — Internet

Figure 2. Server response to *ListMetadataFormats* request



List of Fields

```
header:
  identifier : oai:diglib.ns.ac.yu:19
  datestamp : 2003-10-20T00:00:00Z

metadata:
   dc:
      title: Моделирање и имплементација команди
                      библиотечког информационог система
      title: xml:lang=enModeling and implementing
                      the Library Information System commands
      creator: Biljana Pupovac
      subject: UML, Java, библиотечки информациони
                      систем, команде, радно окружење библиотекара
      subject: xml:lang=enUML, Java, Library
                      Information System, commands, working
                      environment of a librarian
      subject: Моделирање и имплементација команди
                      библиотечког информационог система
      subject: xml:lang=enModeling and implementing
                      the Library Information System commands
      description: Команде у библиотечком информационом
                      систему омогућавају библиотекарима комфорнији
                      начин рада при обради библиографске грађе. Дата
                      је комплетна објектно-оријентисана спецификација
                      команди коришћењем UML-а. На основу ове
                      спецификације извршена је имплементација у Java
                      окружењу.  Поред команди, библиотекар користи и
                      радно окружење, којим се дефинишу основни
                      параметри рада у апликацији. Дата је детаљна
                      објектно-   -оријентисана спецификација Базе
                      радног окружења коришћењем UML-а. На основу дате
                      спецификације извршене помоћу CASE алата
                      RationalRose2000, генерисан је DDL фајл ове базе
                      података. Такође, дат је предлог изгледа
                      екранских форми, које омогућавају ажурирање Базе
                      радног окружења и извршена је њихова имплементација.
      description: xml:lang=enThe commands in the
                      Library Information System provide the librarian
                      with a more confortable method of work in the
                      bibliographic data processing. A detailed
                      object-oriented command specification is given
                      using UML. On the basis of this specification
                      the commands are implemented in the Java
                      environment.  Beside the commands, the librarian
                      uses the work environment to define the basic
                      application work parameters. Using UML a
                      detailed object-oriented specification of the
                      Work Environment Database is given. On the basis
                      of the given specification formed with CASE Tool
```

Figure 3. Server response to *GetRecord* request

3. Implementation of the PMH *data provider* server

As an integral part of the Digital Library of Theses and Dissertations of the University of Novi Sad (details can be found in [8]), the PMH *data provider* server has been developed. The primary task of this server is to allow other participants in the NDLTD initiative, especially their harvesters, to access and harvest metadata from our archive, thus making them available to general public, by the means of the central repository. Use case diagram of this subsystem has been shown in the Figure 4.

In the process of metadata interchange, actors are a PMH harvester at one end, and the Repository Database at the other end of the communicating channel.

As displayed in the Figure 4, the use case **Metadata retreival** includes validation of a client request (use case **Validate Client request**), and gathering data from a local repository (use case **Data retrieval**). In some cases handling of the Resumption Tokens is also necessary (use case **Resumption Token Processing**). Client request validation is neccessary to confirm that the received request is valid in the context of the OAI-PMH, and whether all required parameters for specific action have been received. If that is not the case, the validation process fails and the error code is returned to the user. Error codes are used to allow the client application to determine which part of the request was erroneous. If validation succeeds, the requested action is unambiguously determined, and gathering data from the local repository is performed. If the received action is of a *list type*, i.e. the result of request processing is in the form of the list of individual entities, it may be necessary to perform *Resumption Token Processing* (to prepare the resumption token for the next consecutive request, or to extract the received resumption token from the request). If the request represents continuation of the previous request, the resumption token is extracted, analysed, and if valid, the continuation of the list is retrieved from the database, starting at the last sent item. If the resumption token is invalid, for any reason, in the context of the request, the client application receives an appropriate error code. In that case, the client application can get the full list of responses only by reissuing the complete request.

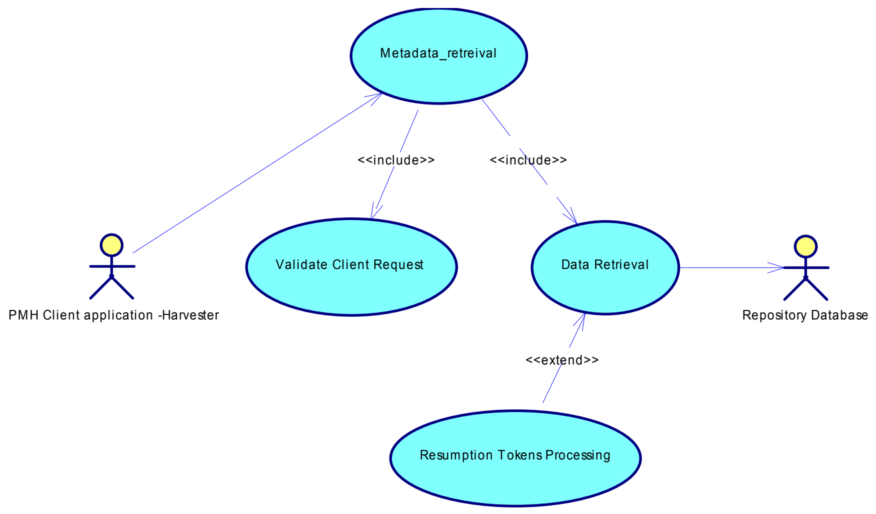


Figure 4. Use Case diagram of the PMH server subsystem

Some institutional repositories have developed their own additional operations that allow client identification, different payment operations for harvesting some metadata, restriction of access rights to parts or whole documents etc. Such extensions are still not standardised and do not fit the current specification of the PMH, and they are only meaningful in the local environments of institutional repositories, while such requests within NDLTD initiative are not processed. The only restriction we have to implement within our implementation is to allow harvesters to “see” only PhD dissertations and MSc theses, since the NDLTD central repository only gathers metadata about these two types of papers. Such selection has been implemented on the level of SQL queries, while graduation theses are still available to other users.

The *PMH Data provider* subsystem, as well as the other parts of digital library, is developed in the Java programming language. This subsystem is implemented as a single java package **com.gint.app.ndltd.oai**. The class diagram of this package is displayed in the Figure 5.

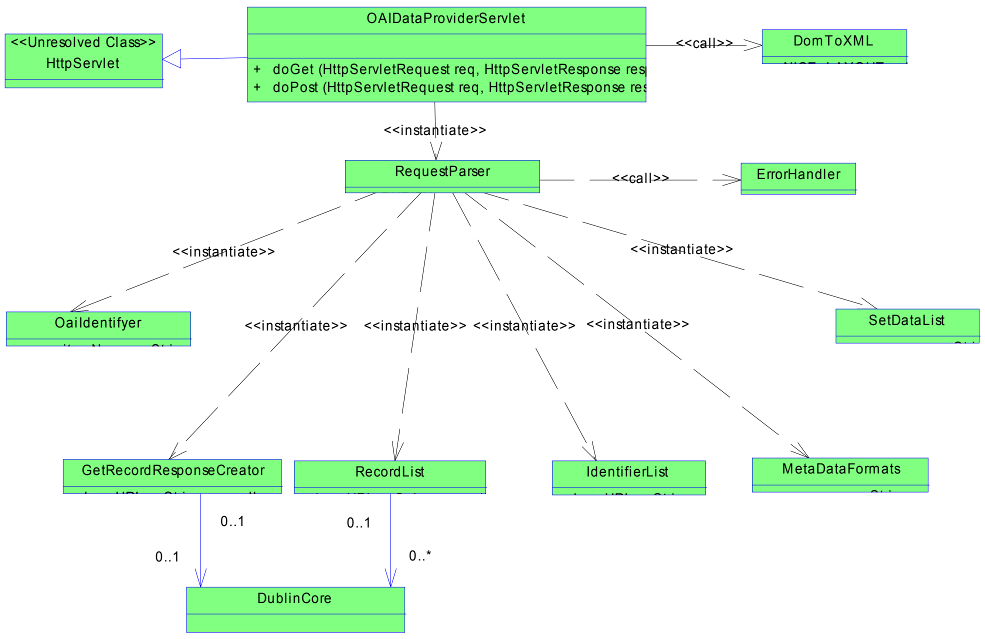
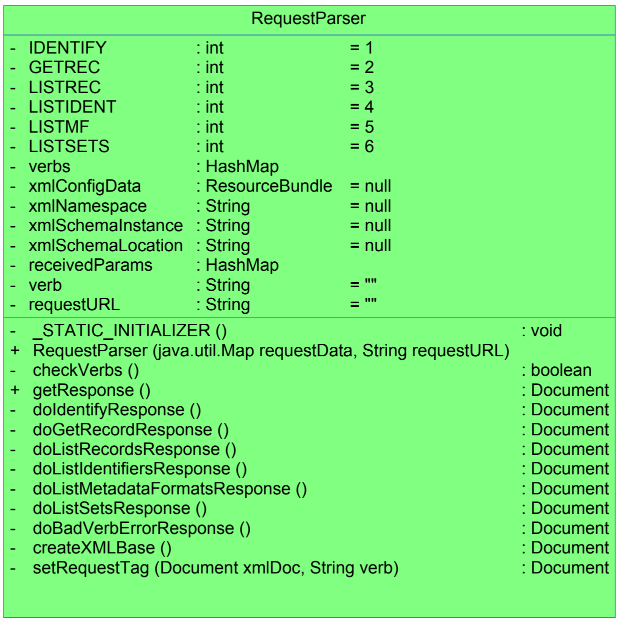


Figure 5. Class diagram of OAI data provider subsystem

As the PMH is a special case of the HTTP protocol, the component which accepts client requests and performs initial processing is the Java servlet **OAIDataProviderServlet** (Fig. 5). Tasks performed by this servlet include the following: receiving an HTTP request, extracting parameters from the request, forwarding these parameters for validation and further processing, and at the end of processing, sending the HTTP response to the client. As specified by the PMH, the response sent by this servlet must be in the text/xml format (or an HTTP error code). Validation of received parameters and their respective values is performed by an instance of the **RequestParser** (Fig. 6), which is responsible for further processing of the client request.

This class has the central role in client request processing (the definition of this class is shown in the Figure 6). An instance of this class checks conformance of a client request with the PMH specification, and depending on the value of ***verb*** parameter determines which action is required to fulfil the client request. Given the value of this parameter additional checking for the existence of possible other parameters and their respective values is performed. If an invalid parameter or/and invalid value, or a required parameter for this type of action is missing further processing is interrupted, and an appropriate error code is inserted into the XML response to the client. Error processing is centralised and is performed by the **ErrorHandler** (Fig. 5).



```
RequestParser

-  IDENTIFY           : int              = 1
-  GETREC             : int              = 2
-  LISTREC            : int              = 3
-  LISTIDENT          : int              = 4
-  LISTMF             : int              = 5
-  LISTSETS           : int              = 6
-  verbs              : HashMap
-  xmlConfigData      : ResourceBundle   = null
-  xmlNamespace       : String           = null
-  xmlSchemaInstance  : String           = null
-  xmlSchemaLocation  : String           = null
-  receivedParams     : HashMap
-  verb               : String           = ""
-  requestURL         : String           = ""

-  _STATIC_INITIALIZER ()                                         : void
+  RequestParser (java.util.Map requestData, String requestURL)
-  checkVerbs ()                                                  : boolean
+  getResponse ()                                                 : Document
-  doIdentifyResponse ()                                          : Document
-  doGetRecordResponse ()                                         : Document
-  doListRecordsResponse ()                                       : Document
-  doListIdentifiersResponse ()                                   : Document
-  doListMetadataFormatsResponse ()                               : Document
-  doListSetsResponse ()                                          : Document
-  doBadVerbErrorResponse ()                                      : Document
-  createXMLBase ()                                               : Document
-  setRequestTag (Document xmlDoc, String verb)                   : Document
```

Figure 6. Class *RequestParser*

If the process of parsing the request has succeeded, the **RequestParser** creates one instance of one of the following classes: *OaiIdentifyer, GetRecord-ResponseCreator, ListRecord, IdentifierList, MetaDataFormats, SetDataList,* which handle a single request. Each of these classes is specialised for handling one task, i.e. to perform processing of one PMH action. The archive database is queried in the process, and metadata relevant to the request are extracted from it. The database connection is established using the *DataSource* interface of the package javax.sql, while the resources specific to the accessed database are determined via the JNDI lookup mechanism. In the process of database access only the database of published papers is queried. This database has its content indexed using the modified TextServer [9] developed for the Library Information System BISIS [10], and its content is available for the general public through the search application. If for any reason metadata cannot be read from the database, while the client request is valid, the proper response of the server is to generate an internal server error and respond to the client with the appropriate HTTP error code 500.

The Figure 7 shows the sequence diagram of processing *GetRecord* request. Beside **verb** parameter, this request requires two additional parameters, ***metadataPrefix*** to define which format the client application expects metadata to be presented in, and ***identifier*** determining which item from the repository should be accessed and its metadata retrieved. If some of these parameters are missing, an error code is returned embedded in the resulting XML document.

Sequence diagrams for other requests are pretty much the same, the only difference is that depending on the type of the client request, **RequestParser** instantiates the appropriate class handling the specified task.

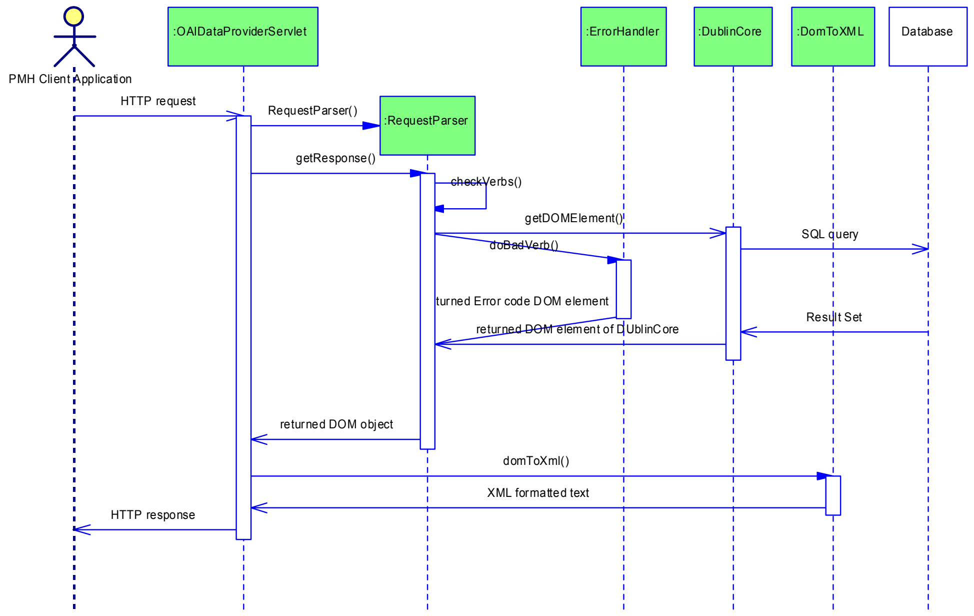


Figure 7. Sequence diagram of *GetRecord* request processing

Currently, this implementation of *OAI data provider* allows metadata to be harvested only in the *Dublin Core* format. Creation of an XML element containing metadata structured as defined by the *Dublin Core* is done by means of the class **DublinCore**.

4. Conclusion

Protocol for Metadata Harvesting is implemented on the server of Digital Library of Thesis and Dissertations of University of Novi Sad, according to the guidelines of OAI. Due to its simplicity, protocol features were relatively easy to implement in given environment. Expansion of this subsystem that would allow metadata exposure in richer formats (UNIMARC, XML format specified for data interchange in Library Information System BISIS) is currently under construction.

Alongside with the PMH *data provider* which enables browsing and metadata harvesting from the repository of the Digital Library, *web service* has been developed to allow interested parties to develop their own client application (or components for larger systems) to access the content of the Digital Library.

References

[1] *Networked Digital Library of Theses and Dissertations*, Dušan Surla, ed., Executive Council of Autonomous Province of Vojvodina, Provincial Secretariat for Science and Techonological Development, Novi Sad, 2003.

[2] Networked Digital Library of Theses and Dissertations – NDLTD. http://www.ndltd.org

[3] The Open Archives Initiative. http://www.openarchives.org

[4] The Open Archives Initiative Protocol for Metadata Harvesting. http://www.openarchives.org/OAI/openarchivesprotocol.html

[5] Santa Fe Convention Base Document. http://www.openarchives.org/sfc/sfc.htm

[6] Dublin Core Metadata element Set, Version 1.1: Reference Description. http://www.dublincore.org/documents/dces/

[7] Open Archives Initiative – Repository Explorer. *explorer version - 1.45a : protocol version - 1.0/1.1/2.0 : March 2003* http://oai.dlib.vt.edu/cgi-bin/Explorer/oai2.0/testoai

[8] Surla D, Konjović Z, Milosavljević B, Sladić G, Protić Z, Komazec S, Okanović D. *An Overview of the Implementation of the Networked Digital Library of Theses and Dissertations.* Infoteka 1-2/2004. pp. 75-86.

[9] Milosavljević B., *Text server for UNIMARC records*, Master thesis, Faculty of Engineering, Novi Sad, 1999. (in Serbian)

[10] D. Surla and Z. Konjović, eds. *Distributed Library Information System BISIS*. Group for Information Technologies, Novi Sad, 2004. (in Serbian).

Integrated Search in the System of Digital Libraries

Goran Sladić, Milan Vidaković
Faculty of Technical Sciences, University of Novi Sad
{sladicg,minja}@uns.ns.ac.yu

Abstract. This paper describes a development of an integrated search in a library network. An integrated search represents searching different library sources, comprised of classical and digital libraries. The BISIS system is used as a representative library software system for classical libraries, while the Networked Digital Library of Theses and Dissertations is used as a representative of digital libraries. User interface for the integrated search is developed as a part of library network portal using portlets. Software agents are used for searching the library network. System implementation is based on the J2EE technology.

1. Introduction

To library network, which consists of the libraries members, users access by web site of library network organized as a portal. Integrated search of library network, one of portals services, is comprised of search of all library information systems in the library network. Service of integrated search enables search of classical and digital library information systems (figure 1). As represent of classical library information systems BISIS [1] information system is used, while as represent of digital library Networked Digital Library of Theses and Dissertations is used [2].

Since integrated search is just one of the services of a library portal, visualization of search is realized by portlet [3]. Usage of a portlet as user interface for integrated search enables easy and efficient integration of this system in a library portal and unique way for searching different library information systems.

One of possible methods for search of the records in network of these systems is by using the agent technology. The agents represent software entities capable for searching and processing the large quantity of information, utilizing a certain degree of intelligence and autonomy. Also, agents offer following things:

- The ability to solve problems that have been beyond the scope of automation – either because no existing technology could be used to solve the problem, or because it was considered too expensive (difficult, time-consuming, risky) to develop solutions using existing technology; or

- The ability to solve problems that can already be solved in a significantly better (cheaper, more natural, easier, more efficient, or faster) way [4].

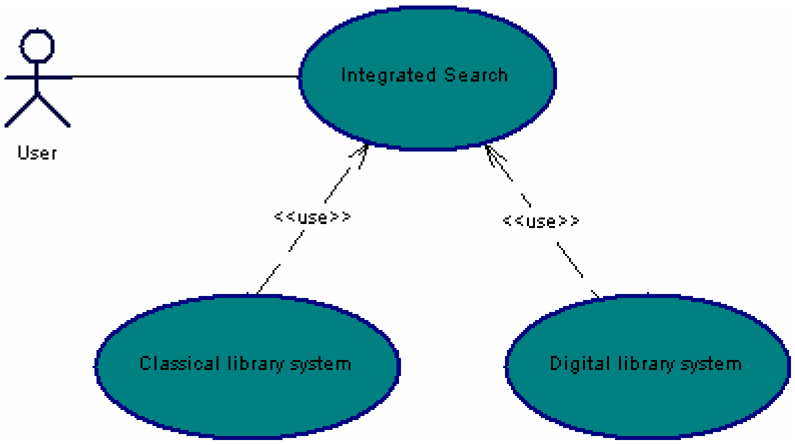


Figure 1. Integrated search of library systems

Agents have to be mobile [5] in order to carry out the job completely. Mobility is the possibility that an agent executes its task on several computers, i.e. it can be capable of “moving” from one computer to another.

In case that there are more different distributed systems with great amount of the data is much more efficient to use the mobile agents, which will “move” form one to another system (computer) and process data locally, than collect data from all systems on one place and process them there. Because a library network represents distributed heterogeneous system the agent’s paradigm is the most natural and consequent approach for implementation of the integrated search of the library network.

Own functions agents executes under the agent center (agent framework), the software system that provide standard system services necessary for the agents functioning. In implementation of the integrated search XJAF (Extensible Java Agent Framework) [6] framework, developed as part of Ph.D. dissertation, is used. XJAF framework is developed according to FIPA specification [7], whereby is enabled collaboration with other agent centers developed according same specification. This enables exchange of information between system of the library network and related systems. Related systems can receive data form the library network system and/or send data to the library network system.

Agent frameworks are hierarchy organized. XJAF framework consists of the managers, which are exchangeable software components (*plug-in*). Each manager executes specific function from the agent center. Concept of *plug-ins* enables adjustment of the framework for specific functionality and environment [6]. In figure 2 is shown concept of *plug-ins* in the XJAF agent framework.

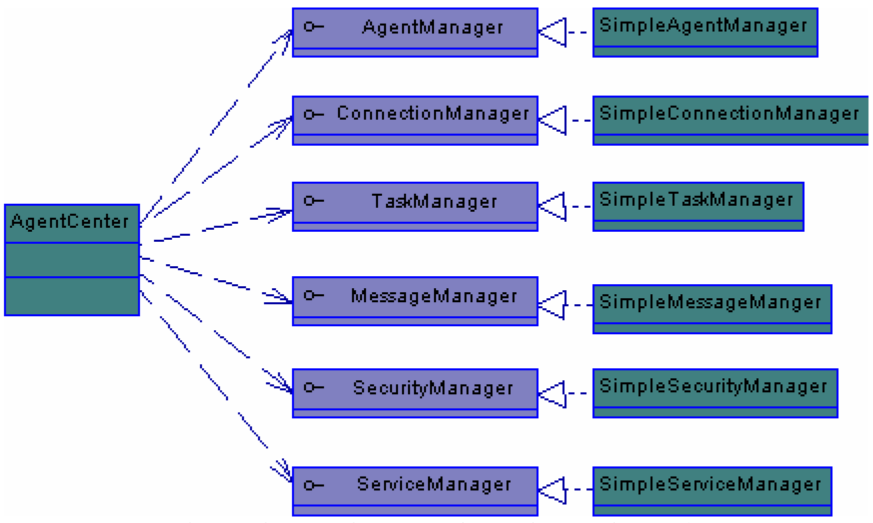


Figure 2. Functions of agent framework are forwards to *plug-in* managers

2. Implementation of Integrated Search

User interface of the integrated search is developed using portlet and as portal server *Jakarta Jetspeed* [8] is used. Portlet is realized using MVC (*Model-View-Controller)* design pattern [9] to separate, much more as possible, showing of content from the program code which generate that content. *Model* component is object that contains result of execution of some action. *Controller* component generates and forwards this objects to *View* component, which shows result. As *View* component is used web page (JSP), which content is shown inside of portlet.

Jakarta Jetspeed portal server has a portlet that uses a MVC design pattern [9]. Appropriate *Controller* and *View* components are set up as configuration parameters. When the user enters data in form and sends them (by clicking on submit button) the particular action of portlet is executed. Result of action is forwarded to appropriate JSP page, which generates a HTML document, which is shown inside a portlet.

Since a library portal is relatively complex system, i.e. each of his portals has complex functionality; model with one action and one JSP page can't provide efficient implementation of requested functionality. For this reason is developed a portlet that provides requested functionality by existing more pairs *View-Controller* components.

Search it self is done by agents. So it is necessary to provide agents with search criteria. Client, in this case portlet, doesn't communicate directly with agent. Communication is preformed via agent center [6].

Client gives task to the agent center and the agent center engaged agent to execute task and returns result of execution to client. Agent center proxy component provides client application to access to the agent center and hides from the client all complex procedures necessary for work with the agents [6]. Client only needs to create the agent center proxy component and forwards the task (search criteria) to it, the agent center proxy then communicate with the agent center. It will send search criteria to agent center and receive result from it. Thanks to this the client (portlet) doesn't need to know for existence of the agents or how they work. In Figure 3 is shown communication between client and agent center.

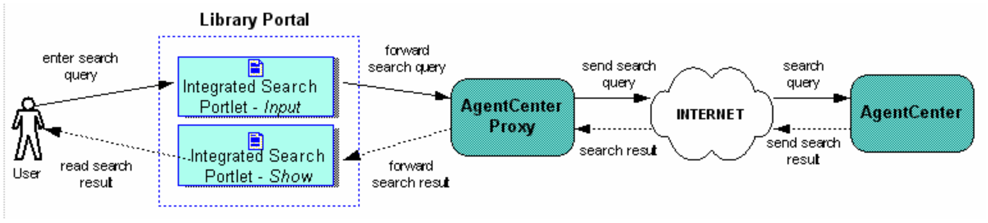


Figure 3. Communication between client and agent center

3. Implementation of Agents for Classical and Digital Libraries

In system of integrated search of library network agent centers are hierarchy organized in two hierarchal levels, although, if necessary, number of levels can be increased. Each agent center handles one library information system. Each center is responsible for search of its library information system, with exception on agent center on top of hierarchy. This agent center communicate with library portal. It sends search query to all other agent centers in system and collect result form them. Figure 4 shows example of hierarchy with three library servers.

Process of adding new library server in system is automated by process of registration of agent center, which handles library information system. Each agent center contains certain number of agents, which are performing search of bibliographic records. In process of search agents are relying on utilize service for searching library server. Two types of agents are developed for search of library network: *IntegratedLibrary-SearchAgent* and *DistributedIntegratedLibrarySearch-Agent*.

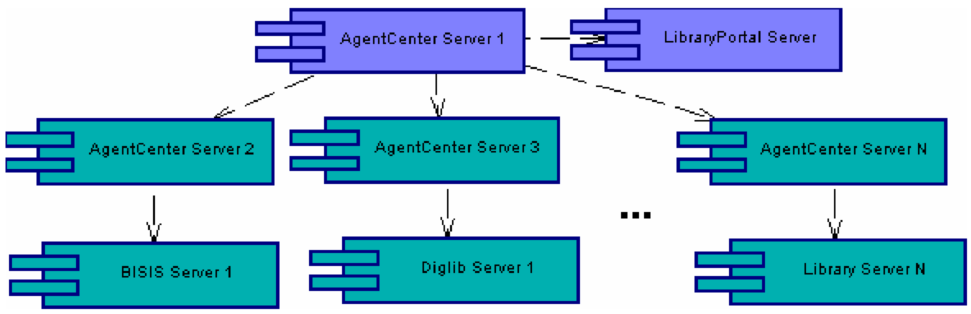


Figure 4. Example of hierarchical organization of agent frameworks and library servers

Search process starts with engagement of *IntegratedLibrarySearchAgent* agent. Engagement of agent means that task which contains search criteria is being forwarded to it. Search criteria are given in XML data format [10]. XML document, which describes search criteria, contains elementary queries and logical operations between them. Elementary query is consists have UNIMARC prefix and value for that prefix [11]. Purpose of this agent is to send search criteria to all agent frameworks in system, take over result from them and forward it back to initiator of search.

IntegratedLibrarySearchAgent starts its job by asking form owner agent center list of all *agent centers* in system. Then it sends search criteria to each agent center. The *agent centers* forwards this search to its agent which is responsible for searching library system (*DistributedIntegratedLibrarySearchAgent)*. This agent uses its local library service to search library information system for which its owner agent center is responsible. Given results and their format (UNIMARC, XML, ...) *DistributedIntegrated-LibrarySearchAgent* agent sends back to *IntegratedLibrarySearchAgent* agent which initiate search. Search of all library networks is shown in Figure 5. Search of local library system by *Distributed-Integrated-LibrarySearchAgent* agent is shown in Figure 6.

As format of query and format of result can be specified it is possible to use architecture of agents search for search according to OAI-PMH [12] protocol i.e. to show result in Dublin Core [13] format of data. On this way portal can be used as PMH *data provider* [12]. Same agents can be used for this type of search because search criteria defined in OAI-PMH protocol can be transform in subset of search criteria defined with UNIMARC prefixes.

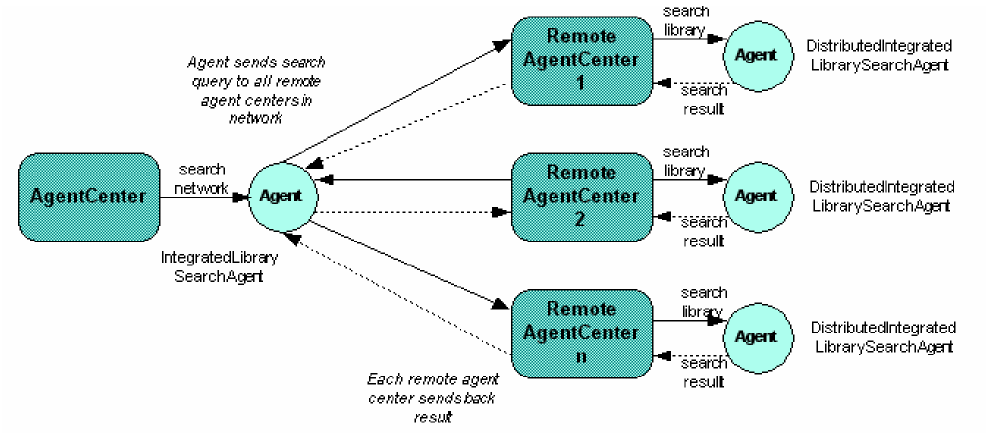

Agent sends search query to all remote agent centers in network
AgentCenter
search network
Agent
IntegratedLibrary SearchAgent
Remote AgentCenter 1
Remote AgentCenter 2
Remote AgentCenter n
search library
search result
Agent
DistributedIntegrated LibrarySearchAgent
Each remote agent center sends back result

Figure 5. Search of all library network

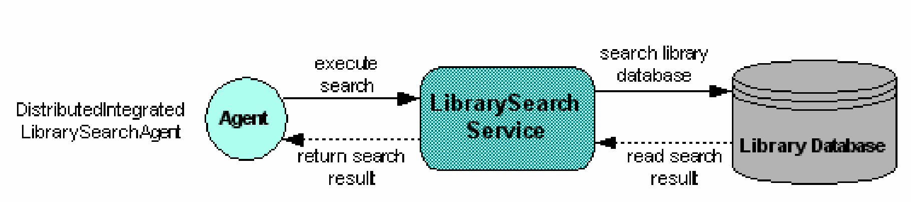


Figure 6. Search of local library system

4. Conclusion

Integrated search of library network comprise search of bibliographic records in different library information systems. As represents of different library information systems BISIS and Networked Digital Library of Theses and Dissertations are used. Inclusion of other library information systems in library network only requires implementation of specialized service for search that library information system.

As component of user interface is used portlet, which supports MVC design pattern. Purpose of portlet is not only to search library network, its functionality depends on actions that are assigned to portlet and can be used for other purpose.

Software agents are used in process of search. XJAF framework is used as agent framework. For search needs are developed two types of agents. To improve search process intelligent agents can be developed. Intelligent agents could decide what library system to search and what not, also they can evaluate of quality of bibliographic records.

Search of concrete library information system is done by specialized service. Services for search classical library information system BISIS and digital library are developed. Future work on services is to develop services for other library systems to support their inclusion in library network.

Same agent architecture is possible to use for implementation of OAI-PMH *data provider* to enable PMH clients to collect data from all library network using one centralized service.

The system implementation is performed using J2EE (Java 2 Enterprise Edition) technology to provide platform independency. All required software is open source what gives opportunity for wide use of this solution.

References

[1] Dušan Surla, Zora Konjović, Biljana Pupovac, Branko Milosavljević, Milan Vidaković, Tatjana Zubić, Tatjana Tošić, *Uputstvo za korišćenje Bibliotečkog softverskog sistema BISIS ver. 3*, Grupa za informacione tehnologije Novi Sad, Novi Sad, 2003.

[2] *Projekat Mrežna digitalna biblioteka doktorskih, magistarskih i diplomskih radova*. (rukovodilac prof. dr. Dušan Surla), Izvršno veće Autonomne pokrajine Vojvodine, Pokrajinski sekretarijat za nauku i tehnološki razvoj, Novi Sad, 2003. http://diglib.ns.ac.yu

[3] *Java Portlet Specification, Version 1.0*, Sun Microsystems, USA, 2003. http://jcp.org/aboutJava/communityprocess/final/jsr168/index.html http://developers.sun.com/prodtech/portalserver/reference/techart/jsr168/index.html

[4] J. Bradshaw, *Software Agents*, (MIT Press: Cambridge, MA, 1997, ISBN 0-262-52234-9).

[5] M. Knapik, J. Johnson, Developing Intelligent Agents for Distributed Systems (McGraw-Hill, 1998, 37-39).

[6] Vidaković, M., "*Proširivo agentsko okruženje bazirano na Java tehnologiji*", doktorska disertacija, Fakultet tehničkih nauka, Novi Sad, 2003. http://diglib.ns.ac.yu/ndltd/docs/set2/ndltd344/MinjaDoktorat.pdf

[7] *FIPA Home Page*, http://www.fipa.org

[8] *Jakarta Jetspeed Home Page*, http://jakarta.apache.org/jetspeed/site/index.html

[9] *MVC Design Pattern*, http://java.sun.com/blueprints/patterns/MVC.html

[10] W3C, "Extensible Markup Language (XML)*",* 2002. http://www.w3.org/XML/

[11] Milosavljević, B., "*Tekst server UNIMARC zapisa*", Master thesis, FAculty of Technical Sciences, Novi Sad, 1999. http://diglib.ns.ac.yu/ndltd/docs/set1/ndltd6/textsrv.pdf

[12] The Open Archives Initiative Protocol for Metadata Harvesting http://www.openarchives.org/OAI/openarchivesprotocol.html

[13] Dublin Core Metadata Initiative http://www.dublincore.org/

Portal of the InterUniversity Library Network

Branko Milosavljević, Srđan Komazec, Zora Konjović
Faculty of Technical Sciences, University of Novi Sad
{mbranko,ksrki,ftn_zora}@uns.ns.ac.yu

Abstract. This paper describes fundamentals for creation of a library network portal. It outlines basic concepts of the portal servers and presents a comparison of existing open source solutions. It then presents basic functions of a library network portal, and identifies portlets, user roles and access permissions.

1. Introduction

A portal represents a web application that provides basic services such as personalization, single sign-on and aggregation of various contents from different sources. It is a presentation layer of a enterprise information system [1]. To achieve simplest possible implementation all portal solutions have a set of basic services. According to the [2], basic services implement these functions:

- customization – users access to the content that is customized according to the their needs,
- aggregation – compilation and presentation of the content of the various types to different users. It takes into account information about the user in the collaboration with the customization and security services,
- support for different device types – preparation of the content for different sorts of the interaction channels (common and cell phones, pagers, fax machines, web browsers, etc.),
- single sign-on – support for the content gathering services to achieve access to the back-end systems and information retreival without need for user to repeat individual authentication and authorization processes,
- administration of the portal – activities that provides correct functioning of the portal
- management of the portal users – categorization of the portal users. Portal presents content to the user in the function of his/her role, interests, location, position, etc.

From the aspect of the user types which have permissions to access the portal and services that portal offers, according to the [2], portals are organized in these categories:

- public portals (like Yahoo),
- enterprise portals (like IBM corporate portal),
- marketplace portals (like eBay) and
- specialzed portal solutions (like SAP portal which provides access path to specific applications).

Common organization of the portal is usually based on visual component aggregation (so called portlets, channels, gadgets, etc.). According to the [1], a portlet is a web component that uses portal services and process requests / generates static or dynamic responses. Content generated by the portlet is called fragment and it has specified structure (HTML, XHTML, WML, XML, etc.). It can be composed with other fragments to build portal pages.

Portlets can be in different presentation states, such as:

- presentation of the content turned on,
- presentation of the content turned off,
- help in the use of the portlet,
- printer friendly view,
- ovelay view, etc.

For every state of the portlet a different interface decoration is applied (normal, closed, help, printer friendly, minimized, maximized, etc). An example of the portal page is presented in Figure 1.

According to the [1] portlet container manages life cycle of the portlets and provides neccesary environment for their execution, like persistent storage for the portlet own use, mechanism for receiving execution requests send by the portal to the portlets associated with specific personalized view and other. The container itself is not responsible for the visual aggregation of the fragments but it delegates that task to the portal application. Portal and the portlet container can be build together or like two separate components of the portal solution. Figure 2 presents schema of a portal.

According to the location of the content that portal access through the portlets it is divided to:

- local content, in the local network (corporative databases, XML data storages, local web services and other). Format of the content is well known and customized for the needs of the organization, and
- remote content, which consists of the informtions and applications on the Internet. Format of the content is customized for delivery through the public network and unknown until delivery time.

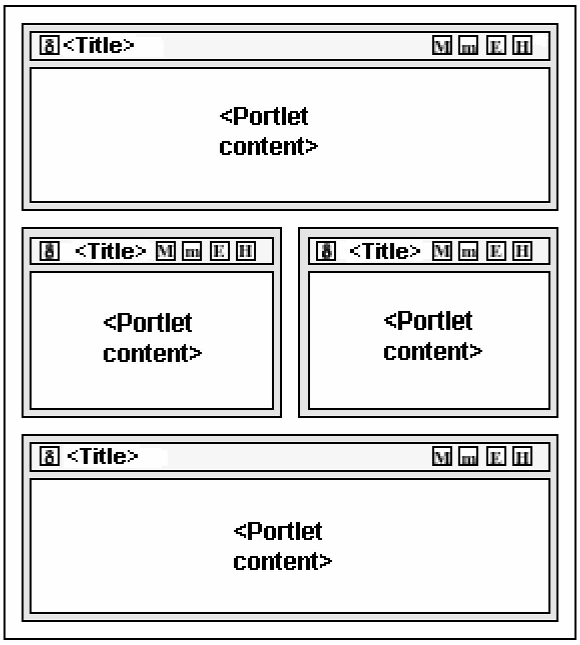


Figure 1. Example of a portal page

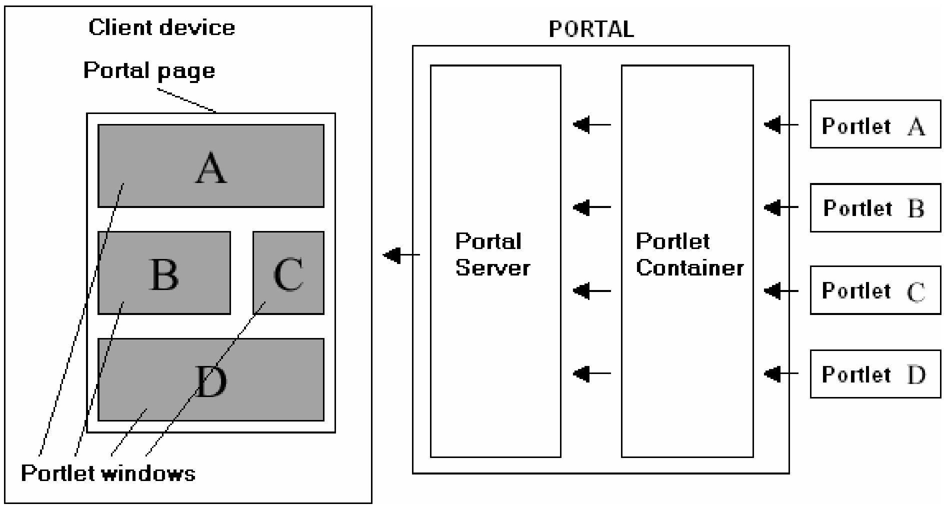


Figure 2. Portal Schema

Content needs to be adapted into the format that the portlets can present. This process can be time and memory consuming, especially when remote content needs to be displayed. Such integration problems can be solved to the speific degree if we apply one of the following techniques:

- publication of the remote content in the format that is supported by the portlets (which minimizes possibilities for the portal server to additionaly change remote content according to the types of the interaction cannels) or
- deploying the portlets on the distribution site. Portal receives content in the specific format and presents it like part of the portal page (which opens the questions in relation to security and stability of the remote distributer).

If second variant is applied, the portal which receives remote content uses generic proxy portlets capable for information exchange with remote portlets (according to the specified protocol). To achieve interoperability between different portal servers and content distributors the Web Services for Remote Portlets [3] (WSRP) standard is used. WSRP component represents a web service that can be pluged in to the portal solution compliant with WSRP or some other web application capable for aggregation of the content from various sources. Portlet proxy components are connected to the WSRP portlets over the SOAP protocol [4]. Figure 3 shows example of the connection of the remote portlet through the WSRP protocol.

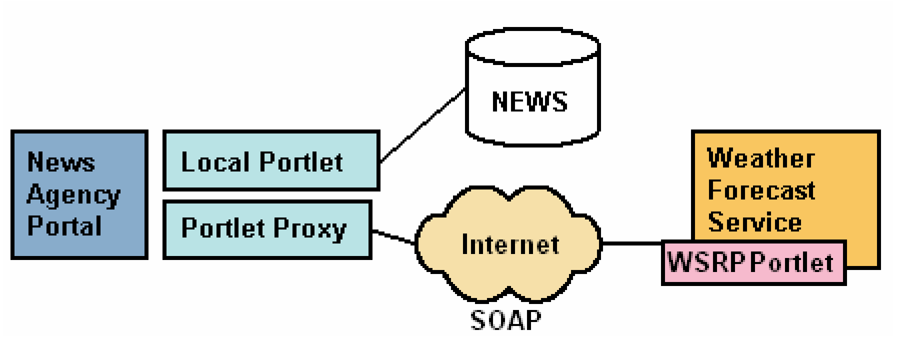


Figure 3. Connection to the remote portlet

J2EE architecture [5] represents a model for creating and managing software components in the environment of the specialized containers (servlet container, EJB container, etc). The portlets are managed by portlet container and communication between portal, portlet container and infrastructure services is standardized by the Portlet API [1]. Typical structure of the J2EE based portal solution is presented in Figure 4.

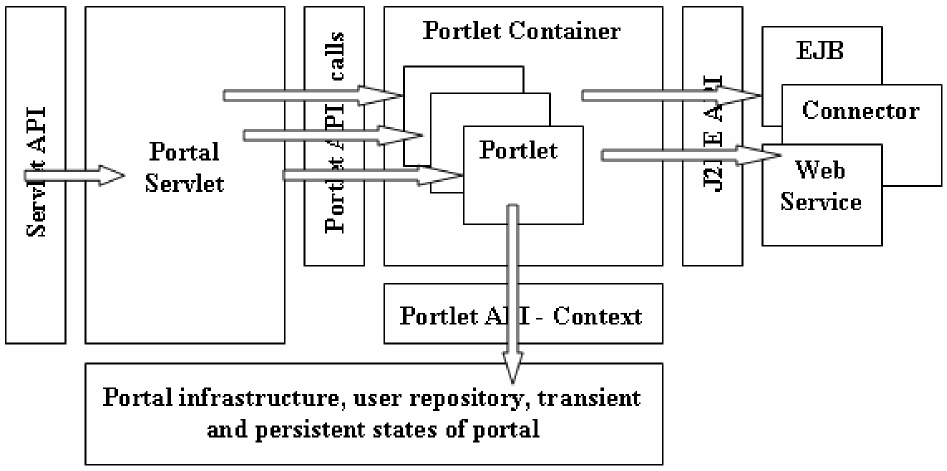


Figure 4. Typical structure of the J2EE based portal solution

2. Overview of Existing Solutions for Portal Implementation

On the portal market there is a lot of the solutions developed by the comercial software vendors (IBM, Microsoft, SAP, Sybase, BEA, etc). This products are complete in such way that they can be easily connected to the existing propietary ERP systems developed in relation to the concepts and technologies provided by respective vendors, they have integrated application servers, integrated content management systems, etc. One of the strategic decisions in the development of the library network software is the use of the open source software in all levels of the implementation. Comparison of the open source portal solutions that can satisfy our requests is presented according to the key criterias such as:

- implementation of the classic infrastructure portal services,
- simple customization,
- flexible API for portlet creation (Portlet API) and
- existance and quality of the documentation.

2.1. Jakarta Jetspeed

Background of the Jakarta Jetspeed [8] portal is composed of XML documents which are used in the content presentation, background functionality, retreival and presentation of the RSS[9] content, presentation of the XML (through XSL) and WML data [10]. Description of the data specific to the portal (presentation styles, personalized informations, portlet registers, etc) is based on Portal Structure Markup Language. There is a large number of developed portlets, capable to process data in various formats (RSS, JSP, Servlet, XML/XSL, Velocity, view of database tables, web pages, etc). Security mechanisms are based on user roles. Security data can be stored in various database management systems. Administrator and user interface can be presented in different languages. Customization of server requires knowledge of not only J2EE architecture but also Jakarta Turbine environment [11]. It is expected soon to be standardized in compliance with JSR-168 specification [12]. Documentation is available.

2.2. Liferay

Beside basic functionality in relation to user and portlet management Liferay [13] has administrative interface, documented support for various DBMS and J2EE applications servers (JBoss, Oracle AS, Pramati, WebLogic, Orion, etc), single sign-on and support for multilingual interface. It is J2EE application based on EJB components and Jakarta Struts[14] environment. Liferay has XML based portlet register and developed various portlets, Jakarta Lucene [15] based mechanism for information retreival. Background API for portlet development is based on EJB specification which is also the gratest flaw of this solution (it is expected soon to be standardized according to the JSR-168 specification). Documentation is available.

2.3. Jakarta Pluto

Jakarta Pluto [16] is the referent implementation of the portlet container based on JSR-168 specification, which means that it provides basis for portlet functionality and leaves the implementation of the portal at minimum level (there are no standard services that can be found in other implementations). Customization is on very low level and documentation is not available.

2.4. uPortal

uPortal [17] represents environment for development of the portal solutions related to academic institutions. It is based on the concept of the channel as information source. It has support for WSRP standard through proxy channels. It supports major DBMS for persisting metadata nedded in portal life. It also supports different types of devices with the high level of customization. There is no support for JSR-168 specification (it is expected soon to be standardized according to the JSR-168 specification through Pluto container integration). Documentation is available.

2.5. Exo

Exo [18] is complex enterprise portal built in modular fashion and based on environments and technologies like Aspect Oriented Content Management

Systems, Java Server Faces [19] interface implementation and other. There is a large number of developed portlets. Because of the high complexity level customization suffers. It supports JSR-168 specification. Documentation is partly available and uncomplete.

2.6. The choice of the Portal Server

After overall examination it is decided that Jakarta Jetspeed (although not JSR-168 compliant) has advantage over other portal solutions in accordance to the specified criteria. Besides good characteristics, Liferay doesn't have expected programming interface, Pluto isn't complete solution and basic concept of the uPortal product doesn't support simple evolution of libray network portal in near future. Exo solution is very complex and documentation unsupported so it doesn't have rapid learning curve which results in prolongued start and end of the portal development. Development of the examined portals is not finished and it is expected to see some other ranking in the future.

3. Library Network Portal Services Specification

Basic goal in library network portal development is to integrate services provided by different library information systems. BISIS library information system [20] and Digital Library of Theses and Dissertations [21] (Digital library) represents two kinds of library information systems – classical and digital library. Integration of their services and creation of new kinds of services based on the composition of existing ones are main purposes of the portal solution.

Figure 5 presents use case diagram of the libray network portal. Use cases have next description:

- user search – metadata records search in one of the classical libraries,
- reservation – reservation of the publications available in the classical libraries,
- records retreival – retreival of the records in some of the standard formats (UNIMARC, XML, etc),
- intelibrary lending – reservation and retreival of the publications in some classical library by the other classical library which are both part of the portal,
- search and retreival of the publications - in one of the digital library which is part of the portal
- author input – of his/her work in one of the digital libraries
- integrated search – of all libraies connected to the portal based on BISIS and Digital library systems

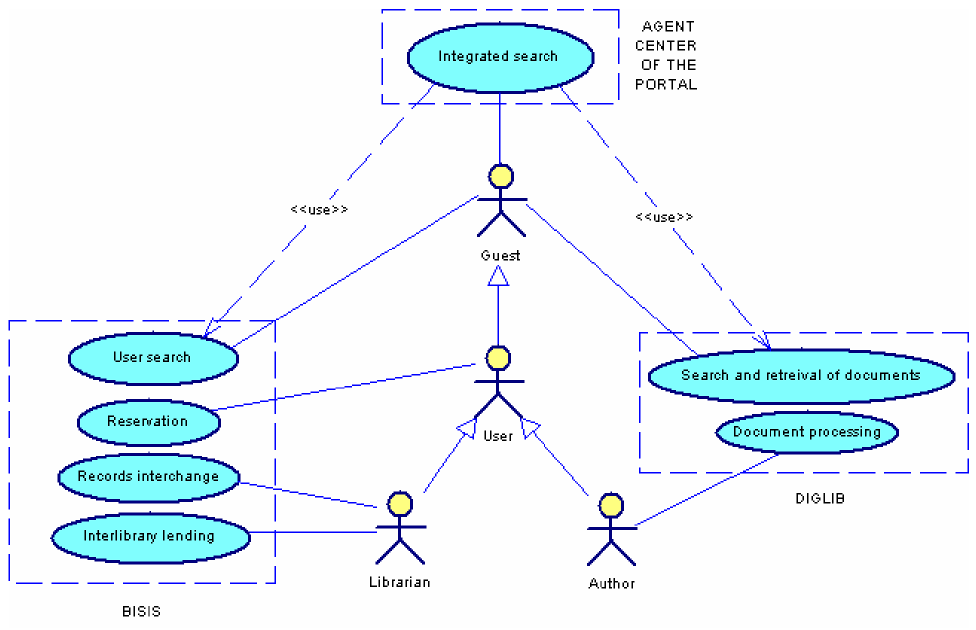


Figure 5. Use case diagram of the library network portal

Service	***Requires authentication***	***Roles with access rights***
Library search and full text retreival	NO	Guest, User, Librarian, Author
Reservation	YES	User, Librarian
Records interchange	YES	Librarian
Interlibrary lending	YES	Librarian
Metadata and full text submit	YES	Author
Integrated search	NO	Guest, User, Librarian, Author

Table 1. User access privileges in the combination with portal functions

Library network portal defines four roles:

- guest of the portal,
- user of the portal,
- author of the work and

- librarian that manages concrete library connected to the portal.

Every role have access privileges defined according to the Table 1.

4. Implementation Characteristics

Turbine environment, as a part of the Jakarta Jetspeed portal, manages user accounts. Four user roles are defined and granted specific privileges from the predefined set of the portal privileges with the permission to change layout of the portlets on the portal pages, turn on and off specific portlets and customize portal pages according to their needs. Jetspeed administrative interface provides ability to simply manage all aspects of user database.

Every use case from the use case diagram of the portal is implemented as a portlet. For implementation details see [22], [23] and [24].

Content of the portal pages are defined according to the user roles and associated portlets. Home page of the portal doesn't require from user to be logged on and provides portelts for searching specific classical and digital libraries and integrated search of all available library information systems (this page is associated to the guest role). To see other pages user needs to be authenticated and authorized.

5. Conclusion

The portal represents web application that provides flexible access to personalized informations and applications from different sources. Portals are based on the aggregation of visual components – the portlets. Portlets can be localy or remotely deployed. Portlets are managed by portal containers. There is a large number of portal solutions on the market, commercial and open source. In the comparation of the open source products in the light of specified criterias Jakarta Jetspeed is choosen for the implementation of the library network portal. Basic functions of the portal are specified by UML use case diagram. For the identified four types of the users access privileges to the portal functions, implemented as portlets, are defined. Althought persented architecture of the portal is developed for the purposes of implementation of library network portal, provided functional analysis and decomposition of the portal are general solutions, independent of concrete portal application, applicable to any other environment.

References

[1] Java™ Portlet Specification Version 1.0. http://www.jcp.org/en/jsr/detail?id=168

[2] Wege, C., *Portal Server Technology*. IEEE Internet Computing, Vol. 6, No. 3, May/June 2002.

[3] *Web Services for Remote Portlets*. http://www.oasis-open.org/committees/tc_home.php? wg_abbrev=wsrp

[4] SOAP Version 1.2 Part 1: Messaging Framework. http://www.w3.org/TR/SOAP/

[5] Java 2 Platform, Enterprise Edition. http://java.sun.com/j2ee/

[6] Schaeck, T., Hepper, T., *Portal Standards*. http://www.theserverside.com/articles/article.jsp?l=Portlet_API

[7] Falkman, D., *Enterprise Portals at Non-Enterprise prices (open-source solutions)*. http://javaboutique.internet.com/reviews/Enterprise_Portals

[8] *Jakarta Jetspeed*, http://jakarta.apache.org/jetspeed /site/index.html

[9] *RSS 2.0 Specification*, http://blogs.law.harvard.edu/tech/rss

[10] WAP Wireless Markup Language Specification. http://www.oasis-open.org/cover/wap-wml.html

[11] *Jakarta Turbine*, http://jakarta.apache.org/turbine

[12] *JSR-168 Portlet Specification Final Release*. http://www.jcp.org/en/jsr/detail?id=168

[13] *Liferay Enterprise Portal*. http://www.liferay.com

[14] *Jakarta Struts*. http://jakarta.apache.org/struts

[15] *Jakarta Lucene*. http://jakarta.apache.org/lucene/docs/index.html

[16] *Jakarta Pluto*. http://jakarta.apache.org/pluto

[17] *uPortal*. http://www.uportal.org/

[18] *eXo platform*. http://www.exoportal.org/

[19] *JavaServer Faces*. http://java.sun.com/j2ee/javaserverfaces/index.jsp

[20] Surla D., Konjović Z., Milosavljević B., Vidaković M., "*Bibliotečki informacioni sistem BISIS ver. 3.01*". Deveta međunarodna konferencija „Informatika u obrazovanju, kvalitet i nove informacione tehnologije", str. 494-504, Zrenjanin, 2000.

[21] Projekat "*Mrežna digitalna biblioteka doktorskih, magistarskih i diplomskih radova*" (rukovodilac prof. Dr Dušan Surla), Izvršno veće Autonomne pokrajine Vojvodine, Pokrajinski sekreterijat za nauku i tehnološki razvoj, Novi Sad, 2003.

[22] Protić Z., Šušnjević T., *Servisi klasične biblioteke u bibliotečkoj mreži*, YUInfo 2004, Kopaonik, 2004.

[23] Okanović D., Sladić G., *Servisi digitalne biblioteke u bibliotečkoj mreži*, YUInfo 2004, Kopaonik, 2004.

[24] Sladić G., Zarić M., Vidaković M., *Integrisano pretraživanje u bibliotečkoj mreži*, YUInfo 2004, Kopaonik, 2004.

Reviews

Experiences of Computer Use in a Scientific Library

Thomas Noßke
Fachhochschule Merseburg, University of Applied Sciences
thomas.nosske@bib.fh-merseburg.de

In 1983/84 first thoughts of supporting the library services by computers were created. At this time the library of Technical College Leuna-Merseburg had no computer of its own like most other scientific libraries in the former GDR. Only the two largest libraries (with around 10 Million books each) had their own computer centres:

- the Deutsche Staatsbibliothek (German State Library) Berlin (founded 1661)
- the Deutsche Bücherei Leipzig (founded 1913, collects all German books)

Both were using Russian UNIX-based computers CM4-20, comparable to the DEC PDP-11.

In Leipzig for all new books data records was created since 1974, but the database were not accessible by users. It was used to create and print the German National Bibliography.

What about the situation of large libraries in the GDR at this time:

- 6 traditional universities: Leipzig (1409), Rostock (1419), Greifswald (1456), Jena (1558), Halle (1694) and Berlin (1809) have traditional libraries with more than 3 million books each.
- 1 universal technical university in Dresden (1828)
- 5 specified technical colleges
  - Freiberg (1765): mining and geo sciences, material sciences
  - Ilmenau (1953): electronics and semiconductor research
  - Merseburg (1954): chemistry, chemical engineering and material sciences
  - Magdeburg (1956): mechanics and construction
  - Chemnitz (in the GDR-time Karl-Marx-Stadt) (founded 1836, technical college since 1963): mechanical and electrical engineering
  - and some smaller technical colleges in Köthen, Mittweida, Zittau, Cottbus and Zwickau.

- In addition to their scientific libraries a large number of medium and larger industrial libraries existed, but most of them had no public access.

To coordinate and increase the quality of providing the scientists in universities and in industry with books and peiodical some specialized networks of libraries were founded. Each of these networks had one university library as the leading unit.

So Merseburg library was leading the network "Chemistry". All other university libraries were members of this network. Also Merseburg library was a member of some other networks.

The chemical industry had the most important and largest industrial libraries of GDR, so the Chemistry library network was the largest network with more than 250 members. Every year Merseburg library has organized some meetings of directors and specialists of the network member libraries.

So the first object of computer use in our library was a catalogue of all chemistry related periodicals (also chemical engineering and biochemistry) in all the network member libraries. It started in 1985. Data records were collected by using an 8-Bit-Computer with CP/M-compatible operating system. This office computer had no harddisc and no floppies, only two tape recorders. The data tapes were sent to the computer centre of our Technical College. There the data were synchronized and sorted. A large report was created and formatted. This process needed two years. It gave our librarians their first experiences in using computers.

In 1987 the Merseburg library was the first university library in the GDR that has hired a computer specialist, but most other university libraries did the same in 1987/88.

In the same year a working group of these specialists was founded under the leadership of the director of German State Library Computer Centre. None of the members had a large amount of experiences with the specialities of computer use in libraries. Twice a year this study group meets in the historical Gotha Research Library for a week to discuss all the problems of computer use in libraries. We also paid attention to the international development of library and information services. We developed our own network of theoretical specialists. The number of practical activities was very low, because the main problem for everyone was, of course, that most libraries had no own computer hardware. All university libraries were dependant on the cooperation with the university computer centres.

In the same year (1987) the Merseburg library got its first personal computer made by ROBOTRON and also a 9-pin dot matrix printer. It was an 8-Bit-computer also based on a CP/M-like operating system. This PC had three 8"-floppy-drives but no harddisc.

We had many problems with some characteristic features of the german alphabet, the diacritics. But also other languages brought similar problems (e.g. with the French, Spanish, Danish, Polish and Czech languages). No existing english language software was able to show, sort and print such diacritical letters. So we started some modifications of our brandnew and very expensiv hardware. The internal character generator 2kByte-EPROM of the alphanumeric display was replace by an 8kByte-EPROM with 4 different self made character sets: english, german, cyrillic and a combined latin/cyrillic set with only capitals. The latter one was very important for online information retreveal in the Russian databases. Also the dot matrix printer got a new EPROM with an extended character set. So we were able to print out our owncatalogue (>500 pages) of the Chemistry library network and distribute it to all members of the network.

In addition to this first computer based activity we tried to establish an online retrieavel centre for our library. An international congress of Polymere sciences 1987 in Merseburg gave us the possibility to establish connections to some important international distributors of scientific online information services. Many Scientists of our university were using printed versions of Chemical Abstracts Services (CAS) in our library and they were looking for possibilities to use the online available service of CAS. In cooperation with the library of the Technical University at Dresden we were able to convince our government of the urgent necessity of Online Information Retrieval in 1988. After very difficult negotiations we got some important permissions. We got money to buy personal computers with modems in Western Europe. We got a specially installed telephone line and we were able to make contracts with Information distributors in Western Germany. As it was absolutely impossible to get telephone lines to Western Germany we made a contract with the Radio Austria Online Service.

In spring 1988 we had first data connections with 2400 Baud to the East German Central Institute of Information and Documentation (ZIID) in Berlin and via this institute to the Soviet Science Information Centre VINITY in Moscow.

To search in russian databases we needed this home made switchable character generator in our 8-Bit-computer. All instructions had to be typed in English, altough all the responses came back in Russian.

After we got a contract with the Scientific and Technological Network (STN International) in Karlsruhe (the European distributor of CAS) we got a special course of online information retrieval in Dresden, held by STN specialists in December 1988.

In January 1989 we were able to establish our first data connection with 300 Baud via Berlin, Moscow, Prague, Vienna and Karlsruhe to CAS in Columbus/Ohio.

A few weeks later we got the possibility to connect directly to RadioAustria/Vienna. But often we had problems getting connected. Sometimes we had to dial several hours until we got a connection. In 1989/90 we made a few hundred retrievals and our researchers got a lot of experiences in this. So in 1990 we were able to establish an STN Online Retrieval Learning Centre in Merseburg that worked up until 1995.

Because of the small number of personal computers before 1990 we had enough time to develop very comprehensive and detailed theoretically concepts of computer use in libraries. After a very short time we recognized the great importance of the most common tool of all librarians and their users: the catalogue.

So we developed not only concepts for catalogue databases and their use but also concepts for collecting data in a library to make an online catalogue available and useful. As we didn't know which structure of bibliographic data would be needed in future we decided to create our own very flexible structure that would make it possible to easily convert this data into every other imaginable bibliographic data structure. Our data only contained latin letters and all additional information in German diacritics were coded by ASCII-character-sets (almost like in HTML). We didn't know when our data would be used in future, so we decided to make very detailed descriptions of our data structure and our own rules of collecting data.

We saw that it was impossible to collect data of existing books and serials without more additional employees. We were able to get some additional employees, but they weren't librarians and they hadn't worked in a library previously. So we had to create a special project to teach these employees to enable them to produce data for our online catalogue in the future.

So our first project was limited in the number of data records and its level of difficulties was low. We had around 20000 dissertations in our library. Dissertations always have just one author, one title and consist of just one part and usually they haven't any additional or complex bibliographic informations. To bring them into our catalogue we created and tested some different concepts. Our three new employees needed around a year to collect all the dissertation data. During this time they learned all the rules of bibliographic data collection. They were taught and supported by our librarians all the time. But our librarians simultaneously have collected all the data from all the newly bought books. So our online catalogue grew up without any interruption from September 1987.

All our experiences of this period we wrote in around 10 articles in some librarian's news-papers. We had a lot of advantages over most other university libraries in the GDR. We were not so big as the old university libraries with their millions of books; we had only 300000 books. We were not so old that we had so much traditional pressures and prejudices. Most of our users were engineers and young scientists who had no problems with modern technologies. So, also, most of our librarians were open to the use of modern technologies.

The majority of university libraries of the former GDR didn't start the process of introducing computers until summer of 1990. In July 1990 (still GDR) our government bought one modern PC with a CD-ROM-drive for each university library. We were able to use it immediately to collect data. Most other libraries at this time started to get their first initial experiences. But they didn't start to collect data then, whereas we had already held data for nearly three years.

All the time we have used a GDR-made software programme that was created to collect bibliographic data. It was called MIDOS (Modulares Informations- und Dokumentations-System). This software is written in the C programming language and was available for CP/M and also for MS/DOS operating systems. We have modified and configured this software to suit our own requirements. The length of each data record was not limited as in the most popular dbase-format. But we also tested a very fast software from the university library at Braunschweig, that is called "**allegro**". This software was also used by the German State Library at Berlin. It was also written in C and at first it was available on UNIX-systems. It was also available for MS/DOS-systems and since 1996 there existed a WINDOWS-interface. A few hundred German public libraries have used the Windows-version right uo until today, as for example in some libraries of the Goethe-Institutes. In our federal state Saxony-Anhalt there still exista an allegro user group of public libraries. Some public libraries in Berlin twice every year link their allegro data records and produces a catalogue on CD-ROM that makes it easy to collect data in other public libraries.

After the end of the data collection of our dissertations we decided to collect data of our existing book stock in addition to the data collection of newly bought books. We thought about the best strategy for tackling this enormous work.

The following possibilities for collecting data we have discussed very intensively:

- in alphabetical order
- in order on the date of acquisition
- depending on the content
- depending on any special characteristics (like dissertations)

- while using a computer based book loan system during the loan process of each book.

At that time books in our library were signed and sorted by the year of acquisition and we didn't have any computer based loan system. We also knew because of our own statistics, that more than 80% of the most popular books were no older than 10 years. That meant that in our library it would be useful to collect data of existing books in the reverse order of their date of acquisition. We started in 1989 to collect data from all books we bought before 1987. By 1996 we were able to complete our data stock for all books we got since 1980. All the time we had the double effort of producing cards for traditional printed catalogue and collecting online data records. BTW the printout of conventional the catalogue cards was based on the data records. For a long time, this was the only use of our computer readable data records.

The political changes of Eastern Germany in 1989/90 brought dramatic changes also to our Technical College and so to our library.

Because of the articles written in librarians' news papers we were known also in university libraries in Western Germany. Most of these libraries had their own personal computers but they never had possibilities to develope concepts for their comprehensive use. Often they only made stand alone projects similar to our dissertation project. Only a few big libraries were collecting data to enable their online catalogues. Most of them had local software solutions and the level of compatibility between their projects was low.

In spring 1990 we were surprised to get invitations to Western German university libraries like Karlsruhe, Oldenburg, Bielefeld and many more. It helped us to establish some very useful connections.

Our perfect preparations made it possible to connect our library to the existing Western German Science Data-Network (WIN Wissenschafts-Informations-Netz) with a special 9600 Baud phone line to Berlin. Only three weeks after German reunification in october 1990 we were online! The main user of this connection was our STN teaching centre. Our hardware had consisted of an PAD (paket assembler-disassembler) that was able to connect 8 PCs via a serial interface. DOS-based communication software PROCOMM was used on our computers.

In autumn 1990 the German government (of the now reunified Germany) started a project for installing personal computers in most important Eastern German university libraries. As part of this project we got our first file server (Novell Netware 3), 5 personal computers (386SX, MS/DOS 3.3) and our first laser printer. So we could install the first local area network in our university in January 1991. In spring 1991 this project was completed by installing a communication server. This server connected our local area network to the commercial data network (DATEX-P) of the German Postal Service by X.25-protocol. It allowed parallel online connections of up to eight computers.

In 1991 very important connections between the four most important libraries (Halle, Magdeburg, Merseburg and Köthen) of Saxony-Anhalt (the German federal state, which includes Merseburg) and the libraries of Lower-Saxony (a federal state in Western Germany) were established. At this time the federal government of Lower-Saxony was looking for a state-of-the-art library software system for all the university libraries in Lower-Saxony (the best known are Hannover, Braunschweig, Osnabrück, Oldenburg, Göttingen and Wolfenbüttel). They have tested all the important existing software systems and found a very promising system in the Netherlands: PICA.

http://www.oclcpica.org

The only apparant disadvantage was, that the PICA system like all existing American systems did not allow the use of German diacritical letters. But modular structure of PICA did allow to creation of modules for other languages. This work was done by the software specialists of the PICA Foundation in Leyden and BRZN (Bibliotheks-Rechenzentrum Niedersachsen = Library Computer Centre of Lower-Saxony) at SUB Göttingen (Lower-Saxony State and University Library) in 1992. In 1991 an agreement was signed between federal government of Lower-Saxony and the PICA foundation in Leyden. The partnership of Lower-Saxony and Saxony-Anhalt and their libraries made it possible to include the scientific libraries of Saxony-Anhalt in this project from the beginning.

At this time PICA was used in all university libraries and all public libraries in the Netherlands. Now PICA found they had got an important partner that was bigger than the whole of the Netherlands and all developments of this system were made together in future.

Since 1993 our librarians have been able to put their bibliographic data on newly bought books directely online to the new central PICA database in the Göttingen computer centre. Our collected data from 1987 to 1993 were brought to BRZN Göttingen. Because of the clever structure and the detailed descriptions of our data records they had no problems in converting our existing data into the PICA data format. So from the beginning our library was one of the libraries with the highest percentage of online available data records in relation to their book stock. Our starting conditions were nearly perfect and much better than in most other libraries that took part in the PICA-project in Germany. We only needed to install a TCP/IP client and a special software on our computers, that was called IBW (Intelligent Bibliographic Workstation). At that time all our software was running on MS/DOS.

PICA in Germany became a very succesful project within a few years. In the period 1993-1995 all scientific libraries of five other german federal states (first Thuringia, then also Hamburg, Bremen, Schleswig-Holstein and Mecklenburg-Vorpommern) joined the PICA system in Göttingen and our own organisation was founded: GBV (Gemeinsamer Bibliotheksverbund = Common Library Network).

http://www.gbv.de

In 1995 more than 180 scientific libraries in 7 federal states were members of GBV. Also all scientific libraries of Hesse (i.e. Frankfurt/Main, Darmstadt, Marburg, Giessen, Kassel) and the two most important non-university-libraries of Germany (Deutsche Bücherei Leipzig and Deutsche Bibliothek Frankfurt/Main) were now using PICA-software.

http://www.ddb.de

http://www.hebis.de

Since 1997 also public libraries in these 7 German member states werea able to become members of GBV.

Meanwhile some other university libraries in German states that aren't member of GBV already have also installed PICA-systems and a large number of university libraries in Scandinavia, Austria and France became members of the PICA-community.

In February 1992, our first CD-ROM server was installed. It made it possible to access expensive CD-ROMs from more than one computer. Hardware consisted of 7 CD-ROM-drives and a small jukebox with one drive and 6 changeable disks. Software was based on a Novell Netware file server and a special CD-ROM server software. The CD-ROM applications were only running on MS/DOS. In May 1995 second CD-ROM-Server was installed. It held 14 CD-ROM drives. In 1996 a new jukebox with 4 drives and 150 places for discs were added. The software was able to run also Windows-applications but management of this system was very expensive and complex. Some of the installed applications needed also new software installations and configurations every monthly database update.

Up to 1995 the PICA system in Saxony-Anhalt was expanded to complete reliability of the first three most important modules in the two local systems. PICA philosophy consisted of three levels. The sole central system CBS (Göttingen) and the local systems LBS (two in Saxony-Anhalt: Halle and Magdeburg) worked with medium sized UNIX-based computers by DEC (in 2002 changed to SUN). The librarians at the third level (all libraries) are worked with IBW-software, at first based on MS/DOS, but since 1998 also 32bit-WINDOWS-versions were available.

The first module is installed in central system and is only available for librarians to collect data. Also the second module was only accessible by IBW, it contains online acquisitions and the ordering of books. The third module is installed in every local system and is also available for users: OPAC (online public access catalogue). At first any catalogue request needed a Telnet-client. The fourth module (online lending system) was installed on PICA level two (local system). Our local system in Halle installed and tested this module in 1998/99.

In March 1996, the Merseburg university computer centre had started its first campus datanet backbone with an Internet-connection. At first our connection bandwidth to the Internet was only 128 KBaud, but the bandwidth has since increased nearly every year in steps of 2 MBaud, 4 MBaud, 8 MBaud up to 34 MBaud. This Internet-connection made possible new services in our library. In January 1996, we have terminated our traditional library catalogue based on printed catalogue cards. From this time we have only had an online catalogue which was one of the most important steps in the history of computer use in our library! In less than 10 years we have changed from a traditional library to an online-library!

We were afraid that our users would have problems in using the online catalogue or would not be satisfied with this new instrument, but we were very surprised that the number of difficulties was really small.

At the same time we prepared our first Internet-homepage. With its help we gave our users information about our library and how to use our OPAC. Users could download free Telnet-software and they got detailed instructions on how to install it and how to access the OPAC from their own PC and from every PC worldwide.

In public areas of our library we made 10 computers available to our users with the Telnet-client for OPAC access and the client for CD-ROM-server access as the sole installed applications. Also two Windows-95-based computers with unlimited Internet-access were now available for our users.

During the next two years we consolidated all existing functions of the PICA-system. In this period also our book acquisition was changed completely to online service. Step by step we have replaced all public computers for just OPAC use with computers with full Internet-access and increased their number to 15 in 1999. We had to recognize that our users often monopoliseded all our free Internet-computers. Sometimes OPAC or CD-ROM requests were impossible because of other users who weren't looking for bibliographic information. They were searching for information on the World Wide Web. So we saw how important Internet-connection was for users in a scientific library and we decided to increase the number of free internet-PCs step by step. But we also needed to give our users the possibility to search our OPAC every time. Often all of our PCs were used and a simple OPAC request was impossible. We bought six special desks with no seat. There users could not sit down, so they only stayed for a few moments to search the catalogue but they didn't search on the Internet. That was a good decision, since now always the OPAC requests are possible.

From 1998 we also prepared the next important step to start using the PICA lending module. Around 50 parameters in more than 5500 possible multi-dimensional combinations are necessary to start the online lending module and must be prepared and uploaded. This work and its complete documentation needed more than one year. Another important condition to start the PICA lending module was to add barcodes to all books. Already in 1996 we started to place self adhesive barcode labels in all newly bought books. The barcodes of all GBV-libraries have the same structure and they include a code that identifies the owning library. This make it possible for any book from any GBV-member to be lent in any other GBV-library. In combination with a catalogue request to the index of all data records in the GBV-system it is very easy to lend out a book from any other GBV-library (inter library loan sysstem). Every year around 5000 books from other GBV-libraries will be lent out in our library and the same number of our books will be lent out in other GBV-libraries.

The test of the online lending module started in October 1999 with a small number of specially selected users. Full service was introduced on the 15th of December 1999. We decided to take this strange date because of many reasons. The last two weeks of December of every year the usage frequency of our library is lower than in November or January. We also wanted to have the possibility of recognizing any problems without any disruption by possible Year-2000-problems. After a small number of inital problems our online lending system has been running prefectely since February 2000.

The standard lending period of books is about 28 days. Of course the librarians can change this period up to 365 days. A number of encyclopedias, dictionaries and other frequently used books are only available inside the library for reference purposes. Such books can be lent out by permission of our librarians from Friday noon to Monday morning. The PICA lending module contains its own calendar, so the system will extend every lending period automatically, if any lending period terminates on a Sunday or holiday. The system is also able to consider every other day when the library is closed.

If the lending period ends and the book is not returned, the user will get a warning and he has to pay a fine. If the user gave us his E-Mail-address initially he will get only a free reminder via E-Mail. If the book is returned within one week, all is ok! But when this time ends without the book being returend of course such users also have to pay the fine.

The OPAC request shows the user a large number of detailed information. It is possible to see how many copies of a book exist and which copy is available. Every book that is in use can be ordered in advance.

Every user has his own account and he is able to access it by web-interface of our lending system every time. By typing in his user number and a password everybody can see which books he has and when they must be given back. He can also extend the lending period up to 5 times by himself, provided thereis no prior reservation by another user. But if the lending period is over he isn't able to extend this time. At first he must come to the library and must pay the fine before the librarian is able to extend its lending period.

In case of lending out a book that isn't yet in the catalogue (maybe it is older than 1980) there is no delay. The book will get a barcode label and so it can be lent immediately. If such book is returned, the librarians will create a complete catalogue data record for this book. So step by step also older books will be available in our catalogue, but only such books that are in use.

In 1999 we decided to replace the existing CD-ROM-server by a modern CITRIX application server. The very expensive hardware was bought jointly the by library and the computer centre. The library offers its CD-ROMs and the computer centre offers software applications that are expensive and not often used like statistical software. So they save a lot of money because only a small number of licenses must be bought. We were able to connect our CD-ROM-jukebox to this multi-processor server (based on Windows NT Server Application Series). The very complex software needs highly qualified administration by two employees, one from the library and one from the computer centre.

Between 1999 and 2004 we have increased the number of our free Internet-PCs up to 36. Six of them are placed in desks without seats solely for OPAC requests. Twelve PCs are placed in two separated rooms where students can work alone or in small groups. There also two scanners and one efficient laser printer are available. Our students are using all these workplaces continuously.

In 2002 we have installed a document server together with our computer centre. It holds all official documents of our university and in future also diploma works. All documents that have to be uploaded must be in PDF-format. To watch any codument you only need a standard webbrowser, but access is limited just to all the computers that are connected to our campus datanet. If you want to access it from outside our university you need to have the possibility to dial our remote access server. Users who want upload a document can find helpful informations in our website. In hard cases of course we will give support to convert existing files in PDF-format.

In 2004 we have installed a new webserver with a content management system (freeware Typo-3). It will be available also for public libraries in our region. Until today we don't have a lot of experiences in using this content management system.

Now our equipment is complete and in future we will improve all our services.

Automation of the Library of Faculty of Humanities and Social Sciences Novi Sad

Gordana Vilotić, Head of Library
Slobodanka Dačić, Senior Librarian
Library of Faculty of Humanities and Social Sciences, University of Novi Sad
gordana@unsff.ns.ac.yu

Abstract. The paper gives an overview and states the significance of the automatisation of the Library of the Faculty of Philosophy in Novi Sad business. Introducing electronic processing of bibliographical material was started in 1991, and the Library Software System BISIS has been in use since 1996. The library was founded in 1954 and there are 550,000 books and magazine editions available in its fund. It is organised in 17 library units – the Centre for Librarianship and Information Activities and 16 seminar libraries – library units specialised in humanities.

1. Introduction

The library information system of the Faculty oif Philosophy in Novi Sad is one of the segments in the teaching and scientific researh process providing various information sources necessary for education and independent work of students, as well as for expert work and scientific research of the teaching staff. Even the Faculty founders were aware of that role of the library at the Faculty dedicating the necessary attention to creating and enabling the library to work at the first meeting of the Executive Board (20th October, 1954). The same attitude has been adopted by all the subsequent managerial structures, thus trying to achieve that the library answers its highly demanding users, both in the size of its funds and in the equipment.

1.1. Library Organisation and Funds

Although the Library went through various phases of organisation for nearly 50 years of its existence, the essence of the conclusions reached on the first Executive Board meeting has never changed: to divide all books according to fields of specialisation "and then separate those with special or for teaching particularly important character and send them to seminar libraries, and store the rest into the Central Library of the Faculty", which today accounts for the library system of 17 units (16 seminar libraries and the Centre for Librarianship and Information Activities).

With the help of the even deposit of funds, a well thought-out policy of acquisition and in the course of time developed exchange with important related institutions in the country and abroad, the Library developed into one of the most important libraries of the humanities profile in Vojvodina, capable of meeting the needs of education and research at the Faculty. By the size of its funds the Library is the second biggest scientific library in Vojvodina (after the Matica Srpska Library), with approximately 550,000 books and magazine editions. The biggest portion of publications is in the enclosed space, stored in the central warehouse and they are delivered to users in seminar libraries and the Central Reading Room by small transport lifts. Textbooks, manuals, major works in various languages of fundamental science fields being studied at the Faculty and current magazines are in the open space in seminar libraries and the Central Reading Room. Beside specialised funds of the seminar libraries, for certain scientific fields the Library has at disposal a rich fund of rarity magazines and books from the 19th century, as well as several important legacies of authors, culture workers and lecturers of this Faculty (Jaša Prodanović, Pavle Stevanović, Đorđe Sp. Radojičić, Stanislav Vinaver, Rudolf Kolarič, Ervin Šinka, Pavle Ivić, etc.).

The Library has more than 6,000 members (5,882 students, postgraduate students, specilising students and 350 teachers and fellow-workers) using annually 120,000 books and other publications. Each year there are 1,000 newly accepted members. The Library users are also students and teachers of other faculties, science and public workers from outside university, students and teachers from high schools, etc.

1.2. Premises and Staff

The seminar libraries are located in 11 rooms situated on two floors. Each seminar library measures between 60m^2 and 80m^2 (totally – app. 950m^2). The Central Reading Room covers 750 m^2, and the central warehouse with offices 1850 m^2. All reading rooms together offer 490 seats for users (40 seats in scientific reading rooms and 450 seats for students). The Library employs twenty one employees (17 librarians and 4 assitant librarians).

2. Business Automatisation in the Library

To the initiative of the Federal Secretariat for Development, the general performer of the System of Scientific and Technological Information of Yugoslavia

IZUM from Maribor installed a computer at the University of Novi Sad in 1991 and connected all faculty libraries to it (by a terminal each). Since the library system of the Faculty of Philosophy is located in various places of the building, the system was reinforced by acquiring an own computer (Mvax 3100) and 10 terminals (VT420), and working on the UCAT, the system of shared cataloguing, started. The system was in operation until May 1993.

The Ministry of Science of the Republic of Serbia financed the development of the System of Scientific and Technological Information of Serbia (SSTIS) from 1993 to 1996. One of the results of that project was the library information system, named BISIS ver. 1.0. The Library of the Faculty of Philosophy accepted the BISIS ver 1.0 system and on acquisition of a server with only three terminals and three PCs put it to the test by entering cataloguing data.

After 1996 the Ministry has not been financing the SSTIS further; however, the system BISIS continued developing at the University of Novi Sad, which resulted in releasing the BISIS ver 2.0 and the BISIS ver.3.0. In accordance with the terms of the contract between the Faculty of Philosophy in Novi Sad and the Rector of the University of Novi Sad's Office, the Library was equipped with the corresponding equipment, the network was augmented and the new system BISIS ver 3.01 was installed in January 2000. Apart from certain minor modifications, this system is still in use today.

2.1. Software

The software was installed in the Library of the Faculty of Philosophy in several steps.

Local shared cataloguing – entering data into the local shared catalogue – and on-line search of the local database were tested in the first phase. Librarians started entering data into the electronic catalogue in compliance with international standards and applying the UNIMARC format for bibliographical data exchange.

The OPAC/user search, a module enabling searching bibliographical material using a Web service, was installed in the second phase.

Local functions of bibliographical material circulation (the system for using bibliographical material) and various types of display from the local database (cataloguing cards, bibliographies, new publications lists, inventory books) were automatised in the third phase.

The software is being used according to the *Manual for Use of the Library Software System BISIS ver.3* (Novi Sad, 2003). The cataloguing processing is done according to the *Handbook of UNIMARC* (Zagreb, 1989), as well as the *COMARC handbook* (Maribor, 1992)

2.2. Computer and Communication Equipment

The minimal hardware requirements for using the BISIS system are:

- for the server a PC with at least 64 MB RAM and a 1GB hard drive;
- for a client a PC with at least 32 MB RAM, a display resolution at least 800x600 and 64 000 colours (Hi color);
- Local computer network.

The equipment satisfying the cited minimal requirements (16 computers with printers altogether) was installed in all Library work premises in the first phase and connected to the server through the old network.

Since the BISIS is a client-server application, the adequate network infrastructure was required. The existing Ethernet network of the Faculty of Philosophy at 10Mb was just augmented and it neither satisfied the system requirements completely nor did it comply with the necessary standards. The problems during work occurred especially when the number of users on the network was large.

In order to increase the quality of work and data security and to achieve the adequate technical development level imposed by the use of the modern applicative software such as the BISIS, the network infrastructure was reconstructed in 2002, both within the Library itself and within the whole Faculty whose part it is.

The network was divided into independent segments merging into the central node. Communication of the segments with the world was achieved through the node, thus enabling the Library segment to access electronic information sources outside the Faculty. On the other hand, it is possible to access data on the Library funds through the WWW service.

The connection to the world and vice versa runs at most at 2.3 Mbps, and the Ethernet segment of the Library itself can support the quality work at most at 10 Mbps towards the server.

Aiming at plannedly and continuously achieving the desired standards, the Faculty plans in the near future to extend its link to the world by supplying new adequate equipment.

2.3. Equipment Today / Hardware Today

Equipment is continuously being acquired using the Faculty financial resources and donations (Table 1), so as to improve the services which the Library offers to users.

No.	Equipment	Quantity
1.	Computers (work)	20
2.	Computers (user)	28
3.	Printers (work)	17
4.	Printers (user)	3
5.	Photocopiers	3
6.	TV sets	9
7.	Satelite dishes	3
8.	Scanners	2
9.	Microfilm readers	2
10.	VCRs	5
11.	CD writers	2
12.	Graphoscope projector	1
13.	Tape recorder	1

Table 1. State of Library technical equipment

All seminar libraries and the Centre for Librarianship have access to the Internet, at least one computer and printer for a librarian, at least one computer and printer for users, as well as additional equipment.

The electronic reading room with 10 computers is located at the gallery of the Central Reading Room and is used by students for the limitless Internet search. It is most often used for producing seminar and graduation papers, for getting extra information and education and for communication by e-mail.

2.4. Librarian Training

In compliance with the Contract signed by the Rector's Office and the Faculty, after the BISIS sytem was installed, the experts of the Organisational Unit ARMUNS have performed the librarian training for work on the system in several groups. The training included the system presentation, using the manual and practical exercises with each group.

Since the librarians already had certain experience working with previous systems, they could work well on this system in a short period of time.

Considering that the main goal of automatisation of library business was efficiently satisfying user demands, the most important task the Library has set to achieve was to make bibliographical records completely electronically available as soon as possible.

3. Database Access

The library today offers end users and lihrarians on-line access to:

- Library electronic catalogue;
- other bibliographical-cataloguing databases in the country and abroad,
- databases and e-publications in the country and abroad.

The quality of educational and scientific work at the Faculty is enhanced to a great extent by access to the Virtual Library of Serbia – an electronic catalogue with approximately 1,500,000 records, foreign academic libraries from Europe and America, foreign databases and the most influential magazines by foreign publishers with full texts, which is financed by the Ministry of Science and Environment Protection of the Republic of Serbia – 12,886 magazines available (EBSCO, Springer Link, Science Direct, ProQuest, Kluwer and others are available through the Libraries of Serbia Consortium for united acquisition).

4. Other Library-Information Services

The library-information services at universitiy sites (electronic catalogues, first of all), contribute to promoting scientific research results in the world and acknowledging the underlying work. These services play the most important role in connecvting faculties and the library to the surroundings and the world, other cultures and systems.

4.1. Library Web Presentation

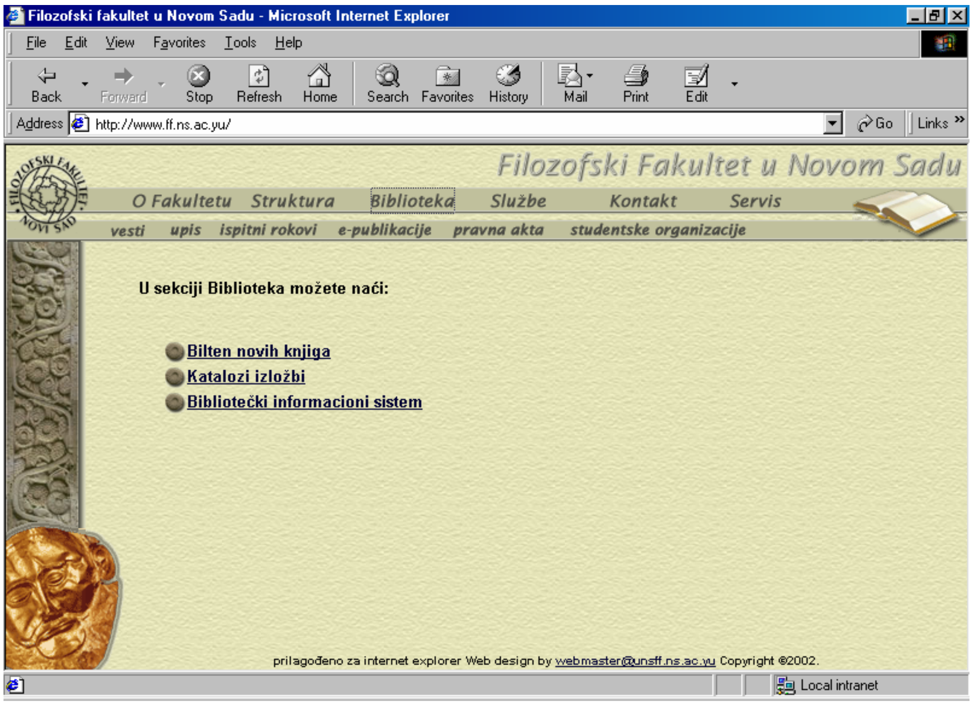
The Library has had its own web presentation since 1998 (Fig. 1) and it is permanently being upgraded and modified (Fig.2). The Library tries to present its funds and services and supply users with information necessary for teaching and research.

The home page of the Library site has three links today.

Web presentation of the New Books Bulletin. As early as in the 1950s the Library of the Faculty of Philosophy started publishing *New books Bulletin*, the publication supplying basic information in the form of a short bibliographical description of the new publications acquired by the Library. The Library attemps to meet all demands made by such a way of presenting material: it recommends the latest issues of books, adds illustrations to material, follows important events related to library-information business in Serbia. The *Bulletin* is published four times a year and distributed to the most important library-information centres in the country.

It has been available in the electronic form since 1998 at the following address: http://www.ff.ns.ac.yu/lokal/bibl/bilteni/index.html.

The Figure 3. shows the Web presentatiom of the *Bulletin* from 1998 which was the only information service of the Faculty site at the time.

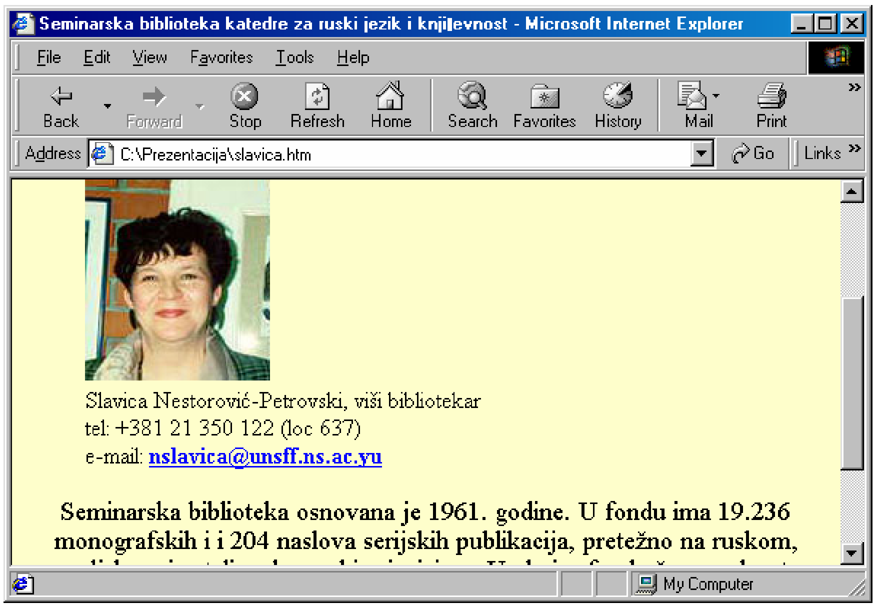

Filozofski fakultet u Novom Sadu - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit
Address http://www.ff.ns.ac.yu/
Go Links
Filozofski Fakultet u Novom Sadu
O Fakultetu Struktura Biblioteka Službe Kontakt Servis
vesti upis ispitni rokovi e-publikacije pravna akta studentske organizacije
Local intranet

U sekciji Biblioteka možete naći:

- Bilten novih knjiga
- Katalozi izložbi
- Bibliotečki informacioni sistem

prilagođeno za internet explorer Web design by webmaster@unsff.ns.ac.yu Copyright ©2002.

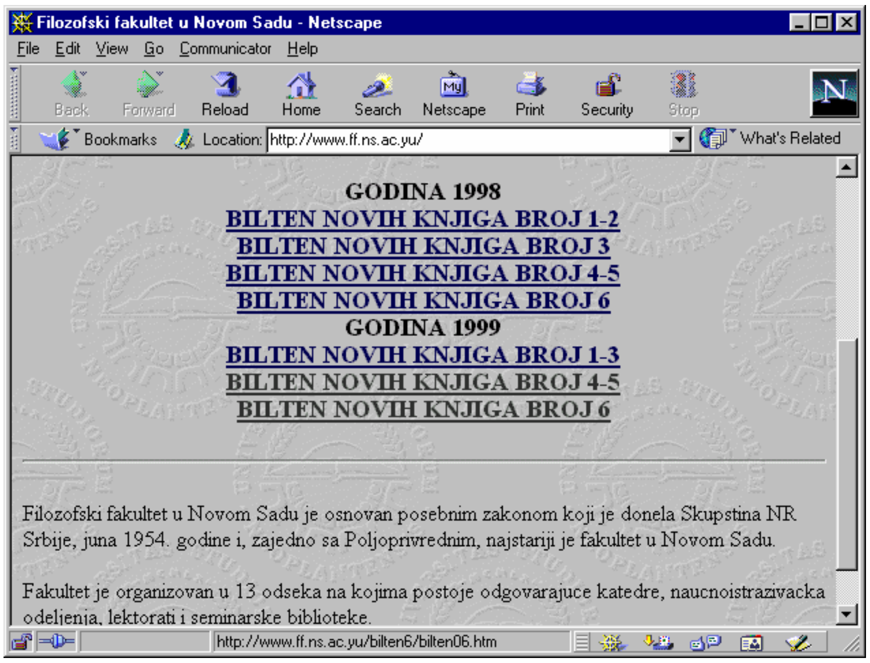
Figure 1. Home page of the Library Web presentation


Seminarska biblioteka katedre za ruski jezik i književnost - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print
Address C:\Prezentacija\slavica.htm
Go Links
My Computer

Slavica Nestorović-Petrovski, viši bibliotekar
tel: +381 21 350 122 (loc 637)
e-mail: nslavica@unsff.ns.ac.yu

Seminarska biblioteka osnovana je 1961. godine. U fondu ima 19.236 monografskih i i 204 naslova serijskih publikacija, pretežno na ruskom,

Figure 2. Seminar library of the Department of Slavistics (under construction)


Filozofski fakultet u Novom Sadu - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks Location: http://www.ff.ns.ac.yu/
What's Related
http://www.ff.ns.ac.yu/bilten6/bilten06.htm

GODINA 1998
BILTEN NOVIH KNJIGA BROJ 1-2
BILTEN NOVIH KNJIGA BROJ 3
BILTEN NOVIH KNJIGA BROJ 4-5
BILTEN NOVIH KNJIGA BROJ 6
GODINA 1999
BILTEN NOVIH KNJIGA BROJ 1-3
BILTEN NOVIH KNJIGA BROJ 4-5
BILTEN NOVIH KNJIGA BROJ 6

Filozofski fakultet u Novom Sadu je osnovan posebnim zakonom koji je donela Skupstina NR Srbije, juna 1954. godine i, zajedno sa Poljoprivrednim, najstariji je fakultet u Novom Sadu.

Fakultet je organizovan u 13 odseka na kojima postoje odgovarajuce katedre, naucnoistrazivacka odeljenja, lektorati i seminarske biblioteke.

Figure 3. Web presentation of the *New Books Bulletin* from 1998

Web presentation of the exhibition catalogue. The Library publishes exhibition catalogues from time to time. The Library in cooperation with Departments presents important authors and marks science and culture jubilees by appropriate exhibitions (Fig. 4).

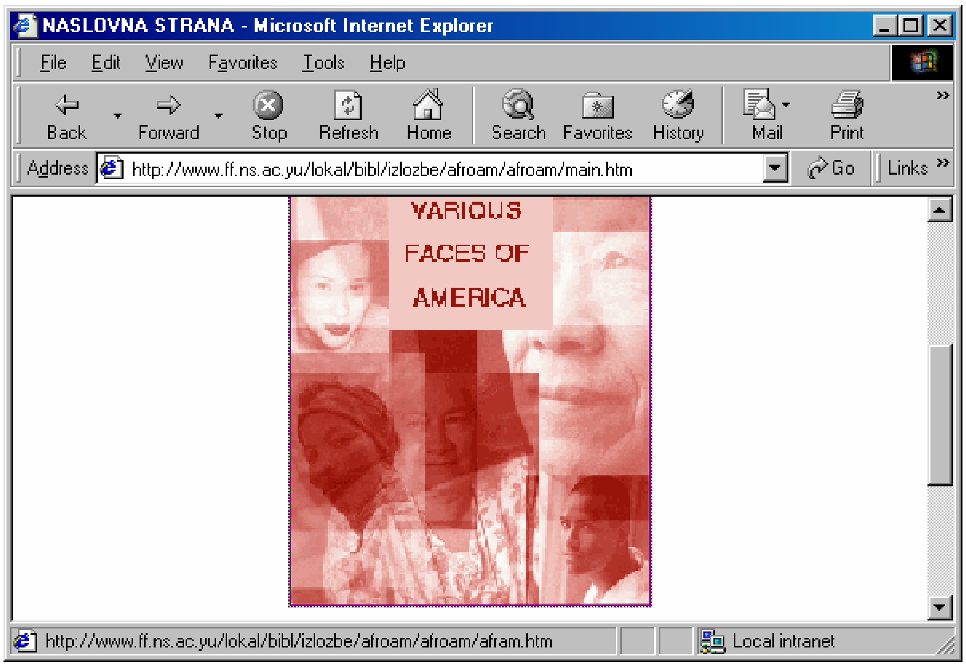

NASLOVNA STRANA - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print
Address http://www.ff.ns.ac.yu/lokal/bibl/izlozbe/afroam/afroam/main.htm
Go Links
VARIOUS
FACES OF
AMERICA
http://www.ff.ns.ac.yu/lokal/bibl/izlozbe/afroam/afroam/afram.htm
Local intranet

Figure 4. Web presentation of the exhibition catalogue

Web presentation of the BISIS. OPAC / user search, the module enabling searching bibliographical material through a Web service, significantly contributes to increasing information potentials of the Library and accessibility of information and bibliographical material. It is intended to be used by a wide circle of the Library users and is available 12 hours from the library and 12 hours from home computers.

The user can choose the language (Serbian and English) in which text messages are displayed and the search mode – a simple query (by a single criterion), or a complex query (containing multiple search criteria) at the site home page (Fig. 5).

Search results are available both as summarised and detailed display.

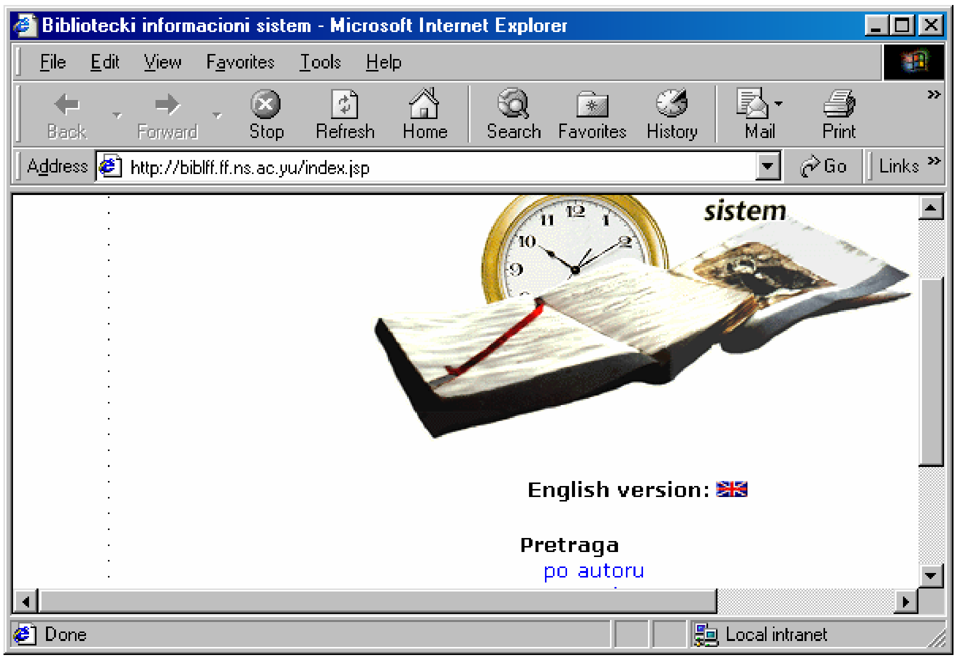

Bibliotecki informacioni sistem - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print
Address http://biblff.ff.ns.ac.yu/index.jsp
Go Links
sistem
English version:
Pretraga
po autoru
Done
Local intranet

Figure 5. BISIS site home page

5. System Maintenance

Until 2004 neither the Faculty nor the Library had a permanently employed expert for maintaining computer and communication hardware. These tasks were performed on call, according to the contract, by an authorised retailer delivering equipment. Today the Faculty has a computer expert taking care of hardware security.

Maintenance of the software system BISIS is, by the contract between the Faculty and the University of Novi Sad, the responsibility of the Organisational Unit Centre for ARMUNS. The contact is kept on a daily basis and every problem is solved within 24 hours of receiving the message signalling a problem.

Supervising record correctness and completeness is performed by the librarian having edited the central catalogue of monographic publications of the Faculty of Philosophy so far.

6. Using the BISIS System and Other Services

One of the important functions of the BISIS ver.3.01 system locally is cataloguing library material in the system of shared cataloguing. The Library network includes 16 client PCs being worked on by librarians.

The main process of the local shared cataloguing in the Library of the Faculty of Philosophy is being executed on the PC of the client, i.e. librarian. The client is connected on-line to the server where the bibliographical material is being stored through the network. The connection of the PC to the server, bibliographical database, is being used while saving a new or modifying an existing record and while searching material only. The system of shared cataloguing is based on cooperative entering and distributed processing of bibliographical-cataloguing data, thus enabling dividing and rationalising work. The existing records in the database are being updated by active users, and a new record is being entered only if it does not exist in the shared database.

The software for local shared cataloguing completely controls entering and modification of database records.

All library-information points in the Library have an Internet connection, enabling end-users to search the electronic catalogue of the Library of the Faculty of Philosophy and bibliographical-cataloguing and other databases in the country and abroad from 8 to 20 hours daily.

The Library electronic catalogue is accessible on the Internet at the address http://biblff.ff.ns.ac.yu.

7. Library Functioning in Conditions of Being Automatised

Due to the Library of the Faculty of Philosophy being located in several places away from each other, entering bibliographical data has elements of shared cataloguing: most data are entered by one cataloguing person, and the rest just add their location data. It is important in view of organising work at the Library, because seminar libraries run their own independent acquisition policy and very often before the same publiucations were processed at several places.

After the test period was over, the Library reached the decision to, starting with the year 2000, enter all aquired books into the electronic catalogue, and to perform processing serial publications and analitical processing of monographic publication in the predetermined order – priority being given to the publications issued by the Faculty and foreign publications. 12,000 records (app. 2% of library material) have been entered into the electronic catalogue so far.

One of the present problems slowing down forming complete and quality local database of the Faculty of Philosophy is the status of the electronic catalogue of the Virtual Library of Serbia. Namely, at this moment national libraries (the People's Library of Serbia and Matica Srpska) which use the software system COBISS do not enable libraries using other software solutions to take over records from the central database formed by uniting databases of the People's Library of Serbia, the Library of Matica Srpska and the University Library “Svetozar Marković” of the University in Belgrade. The coordinated activities of competent ministries which should solve this problem are under way.

The lack of library personnel also presents a problem; every seminar employs a single librarian doing all library activities, from acquisition to informing. His/her duty is to serve users from 9 to 13 hours. Seminars differ in the number of users, in the number of annually acquired books (from 200 to 4,000) and in the fund size (from 5,000 to 50,000 units of library material). An even use of the work hours for entering data in such an organisation is impossible, meaning that organisation will have to undergo certain changes.

The plan of work has not yet been made for the respective conversion. This problem also affects other business segments. Circulation – lending material to the Library users – cannot be automatised, since the electronic catalogue does noty comprise complete material. Librarians enter data on the Library users in the meantime.

The existing software enables printing cataloguing cards, but the Library has not been storing cards into the cataloguing cards catalogue since 2000. Only the electronic catalogue is being updated.

Printing inventory books is not required from all library units, since classical inventory books are still being kept. They will be printed when complete library fund is electronically available.

The development of the BISIS system and the quality of services it offers require that librarians constantly improve knowledge in their field of expertise. The librarians attend innovational and specialised courses organised by the University of Novi Sad. The Library of Matica Srpska, the People's Library of Serbia and the Union of University Libraries of Serbia follow the development of information technologies in the country and abroad.

Thanks to the Union of University Libraries of Serbia, the university libraries using the BISIS system and other currently most often employed systems exchange experience which contributes to improving and unifying library work.

At the same time librarians inform users of the information possibilities of the Library, train them to use all information sources and services and offer their help with finding and evaluating bibliographical data.

Users' training has been given extensive attention since the 1996/97 school year, when the course Informatics was introduced into the curriculum of the Department of Pedagogy and the Department of Psychology. Since the school year 1999/2000 this course is being taught at all departments of the Faculty of Philosophy. This course includes a number of practical exercises to using library information, thus contributing to a more efficient use of information sources and services.

The application of information technologies and various and numerous information offer require that university libraries take over the care of training their users in information literacy. It is understandable, since they are the natural environment where the interests of all participants in teaching, scientific and information processes at the faculty criss-cross – interests of students, lecturers and librarians.

The hardware and software equipment of the Library meet the needs of its users and standards valid for university libraries in the country. The Faculty continuously acquires equipment and does its best to supply end-users with access to all knowledge sources, including communication with local and global library-information databases, because it is a precondition for successful teaching and research.

The Library has provided an organisational means for a free flow of knowledge and information. End-users are offered access to digital and printed information sources 12 hours a day, and the Library electronic catalogue is at their disposal 24 hours a day from their home computers. Students have 18 computers at their disposal for collecting digital information for the purpose of education and an electronic reading room with 10 computers for limitless Internet search.

The Library Web presentations of library-information services enable it to find strategic partners and faster join regional and global processes of integration and they are permanently being upgraded.

8. Instead of Conclusion

The Faculty has supplied hardware and software for teaching, scientific and information activities and provided access to all knowledge sources, including communication to local and global information databases.

Due to the joint engagement of librarians and the software team the level and quality of the software system and the Library services are constantly being improved.

OPAC/user search is constantly being upgraded so as to extend the number of available bibliogarphical data.

The Library tries to make all up-to-date publications electronically accessible.

The process of finding a mode to perform retrospect conversion of bibliographical data with entering the minimal number of data is under way, so as to speed up the process, or to take records over from databases of other libraries. Otherwise, with the current number of library workers and the extensive scope of current tasks, standard data entering would prolong the process indefinitely.

If the Library does not solve the problem of retrospect conversion of bibliographical data in due course, it will be impossible for it to automatise its business.

References

[1] 1. Dačić, S., Vilotić, G.: Definisanje informacione stvarnosti visokoškolske biblioteke kao obrazovne institucije otvorene za inovacije : (na primeru Biblioteke Filozofskog fakulteta u Novom Sadu). U: *Infoteka : časopis za informatiku i bibliotekarstvo*, Beograd, YU ISSN 1450–9687, god. 1, 2000, br. 1, str. 3–17.

[2] Surla, D. et al: Internet za bibliotekare, Zajednica biblioteka univerziteta u Srbiji, Beograd; Prirodno-matematički fakultet, Institut za matematiku, Novi Sad, 1998.

[3] Surla, D., Konjović, Z.: Predlog rešenja bibliotečko informacionog sistema Univerziteta u Novom Sadu, Univerzitet, Novi Sad, januar 1995.

[4] Surla, D., Konjović, Z., i dr.: Upustvo za korišćenje bibliotečkog softverskog sistema BISIS v. 3, Grupa za informacione tehnologije, Novi Sad, 2003.

[5] 5. Vilotić, G., Dačić, S.: Visokoškolske biblioteke Univerziteta u Novom Sadu i njihovo povezivanje u jedinstveni bibliotečko–informacioni sistem : (1990–2000). U: *Infoteka : časopis za informatiku i bibliotekarstvo*, Beograd, YU ISSN 1450–9687, god. 3, 2002, br. 1–2, str. 65–80.

[6] 6. Vilotić, G., Milićević, Lj., Stjepanović, B.: Slobodan pristup znanju i informaciji – između zakona i etike. U: *Intelektualna sloboda i savremene biblioteke : zbornik radova*, Filološki fakultet–Narodna biblioteka Srbije, Beograd 2004. (u pripremi)

[7] 7. Vulić, T., Bilić, G.: Retrospektivna konverzija kataloških listića u format YUMARC, Prirodno-matematički fakultet, Institut za matematiku, Novi Sad, 1997.

Retrospective Overview and Current State of ICT Support of the Library Information System in Macedonia

Stana Jankoska
National and University Library “St. Kliment Ohridski”, Skopje
stana@nubsk.edu.mk

Abstract. The paper contents general information regarding the status, functions and funds of the National and University Library “St. Kliment Ohridski” – Skopje. In more details a chronological survey of activities of the Library in the process of modernization of its functions and Library Information System (LIS) based on ICT is presented. Some of the problems are emphasized too, like the problem of supporting software for library’s functions and libraries network communication infrastructure that is are the conditions for successful development of the LIS.

1. National and University Library “St. Kliment Ohridski”

The National and University Library “St. Kliment Ohridski” – Skopje is a national and state institution of Republic of Macedonia. As a central institution in the library activities, it performs a group of functions: deposit and central library, Agency for ISSN/ISBN, Bibliographic centre, state Centre for librarians education, state Center for development of the librarianship concerning standards introduction and usage, automation of the library processes, a central place of the Library Information System in Macedonia based on the current computer and telecommunications technologies e.c.t.

The main mission of the Library is, across the LIS, its participation in the World information space and its transformation to the state information centre.

2. Retrospective overview and current state of ICT support of the Library Information System in Macedonia

Until the end of 1988 year introduction of computers in the function of performing the library processes in the National University Library (NUL) and the minor part of the public and academic libraries can be characterized as experimental and without system approach. These libraries have bought PC computers and developed or bought some software solutions mostly for processing smaller catalogues. Examples are experiments with union catalogue of foreign periodical publications, catalog of doctoral thesis, scientific projects funded by Ministry of Science and similar.

From the end of 1989 year, NUL was one of the participants in the SNTIJ - System of the Scientific and Technological Information of (former) Yugoslavia Project. Under the framework of the Project NUL was equipped by server with related software and communication equipment as Library application software. IZUM Maribor funded by former Federal Yugoslavia authorities realized project.

Installed ICT solution has enabled:

- NUL inclusion in the SNTIJ library network of its members,
- Link to the World computer networks,
- Information flows in the frame of the SNTIJ project and internationally (through the World computer networks),
- E-mail service (the first usage of the mailing system in the country).

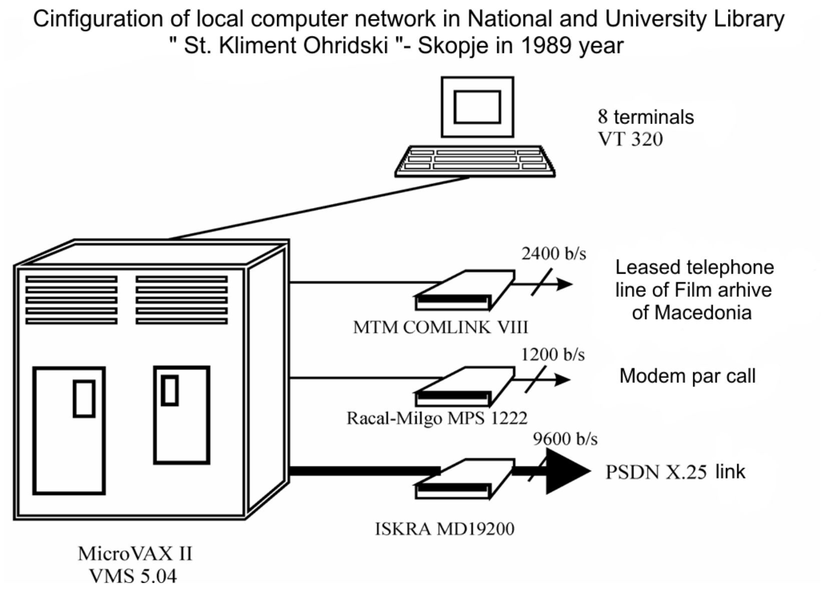

Cinfiguration of local computer network in National and University Library
" St. Kliment Ohridski "- Skopje in 1989 year
8 terminals
VT 320
2400 b/s
Leased telephone line of Film arhive of Macedonia
MTM COMLINK VIII
1200 b/s
Modem par call
Racal-Milgo MPS 1222
9600 b/s
PSDN X.25 link
ISKRA MD19200
MicroVAX II
VMS 5.04

Figure 1. Schema of the NUL LAN in 1989

The version of the Library application software used in the period 1989 to 1995 year has comprised three modules:

- Standardized (UNIMARC) processing of the library materials with specific solutions for the support of the national (Macedonian) Cyrillic scripts and creation of the national bibliographic data base,
- OPAC On –line public access catalog of the national bibliographic data base,
- Standardized forms of the output information products (catalogue cards, national bibliography of monographic, publications and articles).

Advantages of the library application software:

- In this start phase, the application, as one of the base elements of the computerization of the National Libraries included in the SNTIJ, has made a big step in the development of librarianship in former Yugoslavia, as also in Macedonia,
- The standardization of the library processes was upgraded,
- Rationalization as result of cooperative cataloguing on the level of the former Yugoslavia state.

Disadvantages of the library application software:

- NUL did not have source version of the installed software, consequently,
- There was no possibility for own development and improvement,
- There was no possibility of own upgrades in the sense of the support of integrated library system at the level of a library, state or World.
- The maintenance was made by IZUM through the network or locally.

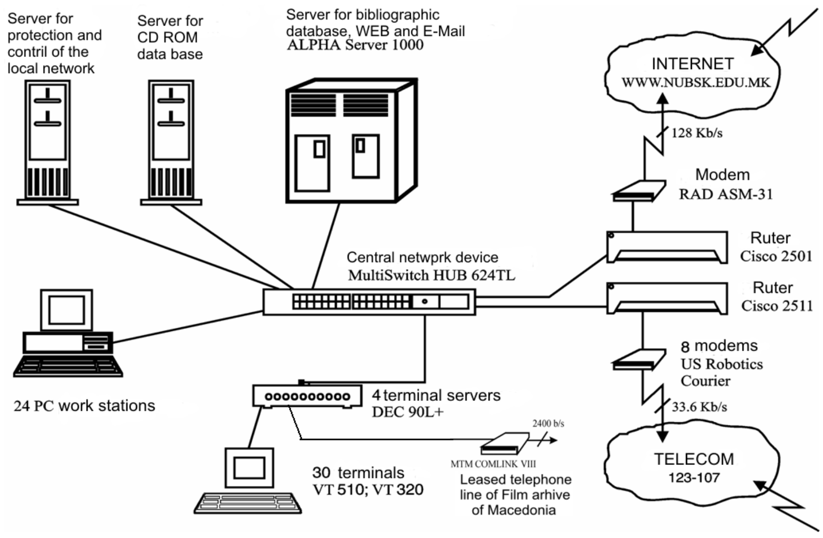

Server for protection and contril of the local network
Server for CD ROM data base
Server for bibliographic database, WEB and E-Mail ALPHA Server 1000
INTERNET WWW.NUBSK.EDU.MK
128 Kb/s
Modem RAD ASM-31
Ruter Cisco 2501
Ruter Cisco 2511
Central netwprk device MultiSwitch HUB 624TL
24 PC work stations
4 terminal servers DEC 90L+
8 modems US Robotics Courier
33.6 Kb/s
2400 b/s
MTM COMLINK VIII
30 terminals VT 510; VT 320
Leased telephone line of Film arhive of Macedonia
TELECOM 123-107

Figure 2. Schema of the NUL LAN in 1995

In the 1995 year, NUL has invested in the new equipment: server, related communications, and the upgrade of LAN, that supported newly established INTERNET connection based on TCP/IP protocol (so replacing older and costly DECNet).

However, what with application software?

The new library application software COBISS, for the standard library processes was installed (with more functionality), due to the fact that technical characteristics of the new equipment did not support the migration of the existing (old) version of application. Consequently, the new contract with the IZUM was signed for using COBISS, but now under leasing conditions. Bibliographic database (with around 80.000 units) was transferred on the new installment

COBISS characteristics:

- Supports the organization of NUL,
- Includes more modules for library processes,
- Has good documentation for users,
- Has organized education (but very expensive),
- Supports on-line help for the users,
- Easy to handle but for librarians who knows the classical standards for processing library materials.

This new solution became very expensive for R Macedonia due to facts that:

- It is offered only under leasing conditions (example: for 200 libraries around 120.000 eur/year)
- Usage of old fashioned VMS operating system
- Specific and expensive hardware and
- Not networked libraries along with expensive telecommunication infrastructure

In the Republic of Macedonia, there are 214 libraries: NUL "Kliment Ohridski" – Skopje (national library), 41 academic libraries (faculty libraries), 31 public, 58 special, 83 schools. In this moment several libraries use COBISS application: National Library, University library of Bitola, 3 public libraries, 5 academic libraries at the Faculty of Natural Sciences and Mathematics of the University "Ss Cyril and Methodius" - Skopje and the library of Macedonian Academy of Science and Art (MANU).

In the meantime, two projects were made (in collaboration with Faculty of Natural Sciences and Mathematics at University "Ss Cyril and Methodius" Skopje):

- The project for development of library information system in Macedonia (LISM) (1995) and,
- The project for building the System for Scientific and Technological Information of Macedonia (SINTIM) based on ICT (1996).

Up to this moment, there is no realization.

As the support to the national LIS under the annual Library Program 1999/2000 of the Ministry of Culture and Ministry of Science NUL, equipment was upgraded with server of larger capacity and some LAN communication equipment.

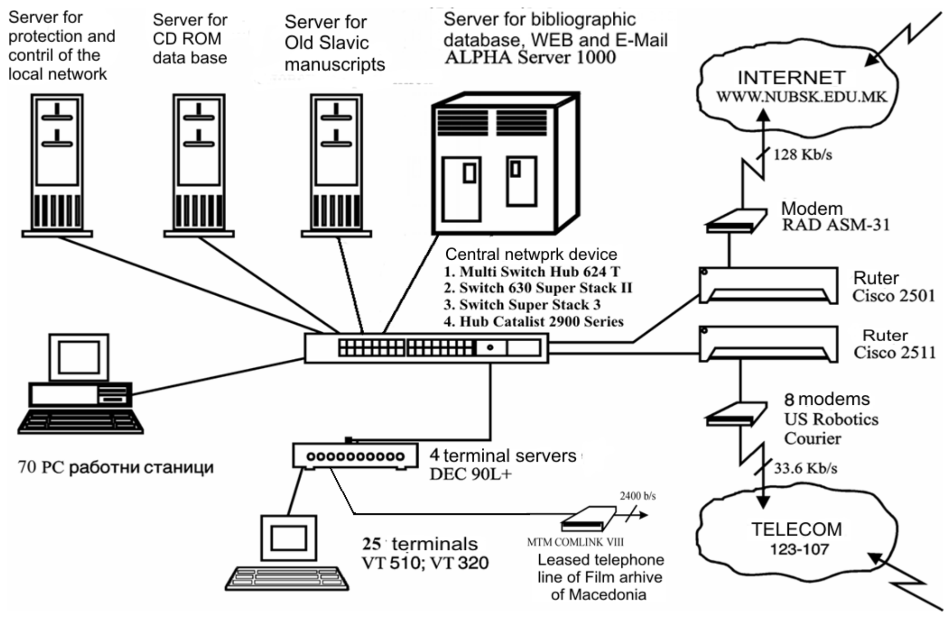

Server for protection and contril of the local network
Server for CD ROM data base
Server for Old Slavic manuscripts
Server for bibliographic database, WEB and E-Mail ALPHA Server 1000
INTERNET WWW.NUBSK.EDU.MK
128 Kb/s
Modem RAD ASM-31
Central netwprk device
1. Multi Switch Hub 624 T
2. Switch 630 Super Stack II
3. Switch Super Stack 3
4. Hub Catalist 2900 Series
Ruter Cisco 2501
Ruter Cisco 2511
8 modems US Robotics Courier
33.6 Kb/s
70 PC работни станици
4 terminal servers DEC 90L+
2400 b/s
MTM COMLINK VIII
25 terminals VT 510; VT 320
Leased telephone line of Film arhive of Macedonia
TELECOM 123-107

Figure 3. Schema of the NUL LAN in 2000

After the equipment upgrade:

- Migration of the bibliographic data base was done from ALPHA 1000 to ALPHA DS20,
- The new version of COBIS was installed with more modules
  - module for sharing cataloguing and
  - circulation of library materials
- Continues the collaboration with IZUM

At the end of 2002 year and on the start of 2003 IZUM – Maribor started the initiative for linking the libraries of former Yugoslavia Republics (newly established states) on the base of COBISS application. The aim is enabling the exchange of the bibliographic information among them and acceleration of the retrospective conversion of the library catalogues formed from 1989 year (NUL has You-deposit units). The NUL is one of contracting sites.

3. Digitization

In the period 1999-2002 year, continually, the Ministry of culture funded three projects for digitization of a part of the NUL Collection of the Slavic manuscripts. Three CD-ROMs containing 3.500 pages of the oldest manuscripts of the Collection were digitized and edited. The database of manuscripts is also available through the INTERNET.

http://www.nubsk.edu.mk

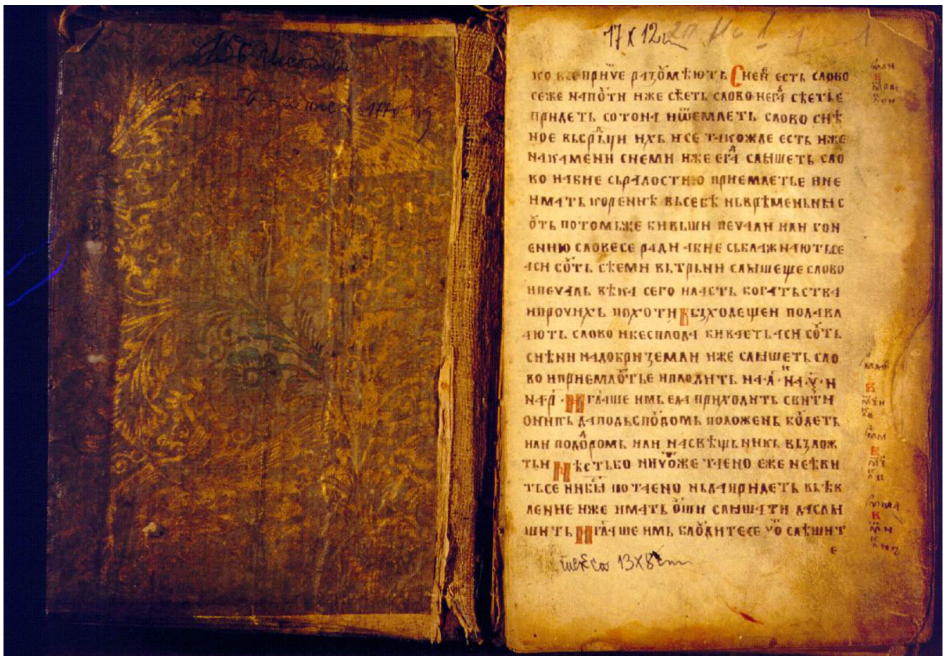


Figure 4. Old manuscript Vrutk, 13th-14th century, pergament

4. Conclusion

- One of the base questions for the functioning of the LIS is the relevant supporting software. Up to now R. Macedonia has no state, system approach to this problem,

- In the absence of the State solutions, the libraries of Macedonia are starting to solve this problem similarly to NUL: they lease COBBIS application,
- Initial installation costs are rather big for individual library,
- Several public and academic libraries are in COBISS cooperative cataloguing but with very slow network interconnections,
- Several connected libraries offer Internet services to its users,
- Initial digitalization work of book cultural heritage has started,
- It is necessary the Government of R Macedonia to invest in the higher speed library network infrastructure and LIS, and by this, to support the librarianship as an important information infrastructure of the education, science, cultural and economical development of the country.