Implementing BRICKS, a Digital Library Management System

Nicola Aloia, Cesare Concordia, Carlo Meghini

Institute of Information Science and Technologies (ISTI)
Area della ricerca CNR, via G. Moruzzi 1, 56124 PISA, Italy
{Nicola.Aloia, Cesare.Concordia, Carlo.Meghini}@isti.cnr.it

Abstract. Digital Libraries (DLs) play a central role in the way information is produced, accessed and used in the Internet era. While the Web provides a universal access to uncontrolled information resources, DLs allow large, distributed repositories of multimedia content, annotated with possibly rich semantic structures and regulated by digital rights, to be created and managed to satisfy the needs of vast user communities. This paper describes the main features of BRICKS, an open-source DL management system based on an innovative peer-to-peer architecture and integrating advanced information management techniques, ranging from model-agnostic content and metadata management to distributed query processing.

Key Words: Digital Library, Distributed Information System, Digital Library Management System

1 Introduction

Digital Library Management Systems (DLMS) [2,3,7,8] are very complex systems, due to their inherently interdisciplinary nature. A DLMS integrates knowledge, techniques and technology from various disciplines, such as hypertext management, information retrieval, multimedia content management, distributed database management, human computer interaction, to name but a few [1]. Informally, a Digital Library (DL) is a *managed* collection of information, with associated services, where the information is stored in digital format, and accessible over a network [2]. A digital library is not really a "*digitized-library*" [3] nor an automatic library management system, although it must offer at least the services of a traditional library with respect to the way information is stored and accessed. An attempt to define a formal model for DLs is in [1], which defines a Digital Library as consisting of a repository, a set of metadata catalogs for a set of collections in the repository, and a set of services for a society of users. A collection is a set of digital objects. A digital object can be either a simple stream or a complex composition of multimedia objects. The services of a DL must include indexing, searching and browsing of the stored objects. The society of users is the highest level component of the DL, including information needs, use, privacy, ownership, intellectual property right, security, and other user-related concepts.

BRICKS is a DLMS developed in the context of the European Integrated Project: *Building Resources for Integrated Cultural Knowledge Services,* in the Sixth Framework Program[1], under the IST priority[2]. The aim of the BRICKS project has been to design and develop an open-user and service-oriented infrastructure to share knowledge and resources in the Cultural Heritage domain. Even if the primary goal of the BRICKS project has been to supply infrastructure and suitable services for the Cultural Heritage community, with a special emphasis on small museums, its resulting DLMS can be successfully used in different application domains. BRICKS integrates into a unique, coherent design, a number of relevant enabling technologies, making them available as Web Services through simple but powerful Application Programming Interfaces. This paper presents the main features of BRICKS, focusing on the Collection Manager. The BRICKS Collection Manager[3] provides the operations for managing collections and their content, and for querying the information space, making distribution and heterogeneity transparent [11]. The paper is organized as follows. Section 2 gives a brief overview of the relevant literature as well as of comparable DL management systems. The basic BRICKS functionalities are described in Section 3. The BRICKS component-based software architecture, along with a brief description of the main components, is presented in Section 4. Section 5 contains a detailed description of the Collection Manager (CM) component. Conclusions and future works are outlined in Section 6.

2 Related Work

The notion of "*a digital library seems hard to completely understand and evades definitional consensus*" [1]. DLs seat at the cross of many roads. Christine Borgman [6] identifies two distinct senses in which the term "digital library" has been used:

- The technological definition stating that "digital libraries are a set of electronic resources and associated technical capabilities for creating, searching and using information". In this sense DLs are an extension and enhancement of information storage and retrieval systems that manipulate digital data in any medium (text, images, sounds; static or dynamic images) and exist in distributed networks. This view is contrasted by:
- the social view stating that "digital libraries are constructed, collected and organized, by (and for) a community of users, and their functional capabilities support the information needs and uses of that community". In this sense DLs are an extension, enhancement, and integration of a variety of information institutions as physical places where resources are selected, collected, organized, preserved, and accessed in support of a user community: libraries, museums, archives, schools, offices, laboratories, homes, public spaces, and so on.

[1] www.brickscommunity.org

[2] Contract no. 507457

[3] The BRICKS Collection Manager has been designed and implemented at the Institute of Information Science and Technologies of the Italian National Research Council (CNR).

Historically, DLs were launched around the mid 90s by the U. S. Digital Library Initiative in two successive phases, aimed at "dramatically advance the means to collect, store, and organize information in digital forms, and make it available for searching, retrieval, and processing via communication networks -- all in user-friendly ways". As it is well-known, Google was founded by researchers recruited for the first DL Initiative. The European awareness on DLs started in the 5th Framework Programme and continued in the 6th. In the context of these Programmes, many projects were funded, sided by the scientific activities carried out within the DELOS Network of Excellence on DLs[4]. Significant activity has also been carried out in non-US, non-EU countries, notably New-Zealand[5].

The notion of Digital Library Management System (DLMS) is still not standardized. Currently, there exist software platforms supporting operational Digital Libraries, each having its own functionality with some overlapping and practically no agreement on a common information model. In fact, the most substantial and systematic attempt at creating a reference model for Digital Libraries is an on-going activity of the DELOS Network of Excellence [10,13,18]. Relevant on-going projects:

- Massachusetts Institute of Technology (MIT) and Hewlett-Packard Labs develop the DSpace[6] [14,15] digital repository management system. It is used by many organizations around the world to store, preserve and disseminate scientific and educational content.
- The Fedora[7] [16] Management Service defines an open interface for administering the repository, including creating, modifying, and deleting digital objects, or components within digital objects.
- The DILIGENT[8] [17] project is creating an advanced test-bed that will allow virtual e-Science communities to share knowledge and collaborate in a secure, coordinated, dynamic and cost-effective way. The DILIGENT test-bed will be built by integrating Grid and Digital Library technologies.

Relating BRICKS to each one of these initiatives cannot be done for reasons of space. In synthesis, BRICKS is currently the only project aiming at providing a full DLMS functionality on a distributed infrastructure, exploiting the peer-to-peer paradigm.

3 BRICKS Functionalities

BRICKS is an extensible Distributed Digital Library Management[9] built as a set of web-services (BRICKS *foundation services*). The foundation services provide generic support for basic DL services, such as storing, querying and accessing both digital

[4] www.delos.info

[5] www.nzdl.org

[6] www.dspace.org

[7] www.fedora.info

[8] www.diligentproject.org

[9] published as open source GNU LGPL (http://foundation.bricksfactory.org/)

content and metadata; integrating heterogeneous metadata; for security services for authentication, authorisation and digital rights management. BRICKS code is platform independent and scalable.

The main goal of BRICKS DLMS is to supply the functionality for helping institutions in (a) setting up their digital libraries and (b) implementing sophisticated applications on top of the global content network, based on a distributed architecture. The complexity of distributed information management (based on a P2P infrastructure in our case) is completely hidden to the application developers. The fundamental BRICKS components provide services for information storage and access, as well as for ontology and metadata schema management. One BRICKS component implements distributed information retrieval capabilities by simple search (Google like), and by advanced search based on knowledge represented by ontology and metadata schemas. Others BRICKS components implement cross-language discovery, possibly personalized based on user profiles. Security management and Intellectual Property Protection, as well as a sophisticated Annotation management component are supplied to users applications without any burden on developers. BRICKS is an extensible system, new components and new functionalities to existing components can be easily added without any change in the already implemented applications.
BRICKS is entirely written in Java and integrates some open source systems for basic functions (e.g. Lucene[10] as a full text information retrieval system). Institutions installing BRICKS software can decide to use proprietary software (e.g. commercial DBMS as a back-end system for BRICKS metadata manager or content manager) when configuring his own environment. Typical digital library tools, are the core applications that can have benefit in using BRICKS, especially those dealing with multimedia documents. Nevertheless cooperative working applications can have great advantages in using BRICKS functionalities.

3.1 The BRICKS Conceptual model

The BRICKS Conceptual Model (see Figure 1) is, at heart, a simple one. A BRICKS DL consists of a set of nodes, called BNodes. A BNode is an abstract notion representing an installation of the system, and it is not specifically related on a single machine: for scalability reasons, a BNode may be deployed on several machines, and for organizational reasons a single machine may host several BNodes, each corresponding to the resources of, *e.g.* one institution. A Bnode consists of several collections, each of which may be of one of three kinds:

1. Physical Collection, structured in sub-collections
2. Virtual Collection, also structured in sub-collections
3. Stored Query.

These notions will be illustrated in detail in Section 5. DL Items are stored into Physical Collections, Each DL Item has a Content Model which describes it from a

[10] http://lucene.apache.org/

structural point of view, that is by describing the part of which it may be composed, the type of these parts and their structure, recursively. It is important to notice that BRICKS does not make any commitment towards a specific content model. Rather, it allows different content models to coexist, each expressed as a model in the Java Content Repository (JCR) API[11]. In the BRICKS conceptual model, JCR plays the role of a content metamodel. In the course of the BRICKS Project, the Docbook[12] and TEI Lite[13] models have been expressed in JCR, as well as other application specific models.

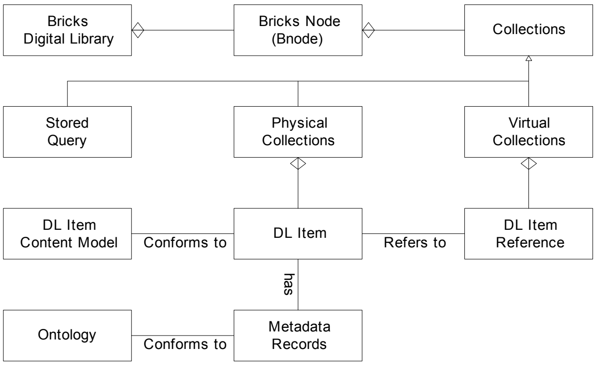

Fig. 1. The BRICKS Conceptual Model

Each DL Item has associated a set of metadata records, which give information about the item required to support several tasks, notably discovery and management. Each metadata record conforms to an ontology, which is expressed in OWL[14]. Analogously to content modelling, then, BRICKS does not make any commitment towards a specific metadata model, but rather supports the coexistence of any model, as long as it can be expressed in OWL. For basic interoperability reasons, all DL Items must have an associated Dublin Core metadata record.

The BRICKS Conceptual Model has other entities, to support fundamental DL management operations, such as Digital Right Management, User Management and others. The model illustrated thus far covers however the main aspects and is sufficient to describe the main aspects of the BRICKS architecture, which is done in the sequel.

4 BRICKS architecture

Figure 2 shows the main components of the BNode architecture. The basic layer is provided by the software implementing the network protocols and the P2P layer. The latter implements the P-Grid distributed hash table [4] [5], that is a hash table whose

[11] jcp.org/aboutJava/communityprocess/final/jsr170/index.html

[12] www.docbook.org/specs/cs-docbook-docbook-4.2.html

[13] www.tei-c.org/Lite/

[14] www.w3.org/TR/owl-features/

entries are distributed among different peers (we will use terms peer, node and BNode as synonyms). Each peer is responsible for a partition of the overall key space and maintains information (routing table) to route requests to other peers to be able to forward requests that cannot be answered locally. On top of the basic layer, there are the BRICKS components proper, categorized in 3 main classes:

1. Fundamental bricks are those present in any installation, because they provide necessary services for a BNode to function.
2. Core bricks are those providing a first, elementary level of service, through which a user can be defined and log in (User Management), be recognized (Authentication and Authorization) and perform content discovery or browsing (Collection Manager).
3. Basic bricks are those completing the functionality of the foundations for full-fledged BNodes wishing to upload content protected by digital right management, to associate metadata to content, also as annotations, and creating more advanced services by composing more basic ones.

The P2P paradigm must be understood in this light: by supporting self-maintainability, autonomy and total freedom of joining and leaving the DL. This paradigm is ideal for those institutions, such as small museums, which want to avoid major investments and heavy commitment.

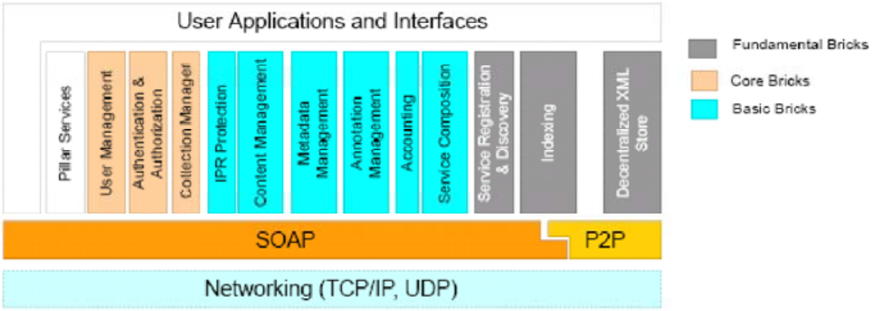

Fig. 2. The BRICKS BNode Architecture

5 Collection Manager

The BRICKS Collection Manager (CM) is a mediator between the DL applications and the DL contents. Applications may range from GUIs to arbitrarily complex programs for performing domain specific tasks. DL contents are the digital objects stored in the DL and their associated information (metadata). The task of the CM is to represent the DL contents via a conceptual model that is understandable and intuitive from an application viewpoint and to offer primitives for the manipulation of this model in support of applications. The CM support two basic kinds of collections: Physical and Logical. Physical collections are a central notion in the BRICKS content model: not only content is organized by physical collection, but so is the discovery of resources and the definition of logical collections. A physical collection is a set of content items (DL objects and metadata) which belong to the same content provider and are homogeneous from the provider point of view (e.g. items are of the same kind, are described by the same metadata format(s), have same digital rights, and so on). A Logical Collection in BRICKS can be seen as a DBMS view. The items of a Logical Collection can be dynamically computed at run time (if we define the collection as a *Stored Query*) or can be materialized as a set of references to DL items. Any authorized BRICKS user can create and operate upon Logical Collections. Once a logical collection is created, it becomes part of the digital library information space and can be searched by other users as any other digital library resource. A BRICKS content item may be referenced in many Logical Collections (in local and/or remote BNode(s)). Conversely, a Logical Collection may contain references to many items, coming from many different Physical Collections stored in local and/or remote BNode(s). (fig. 3).

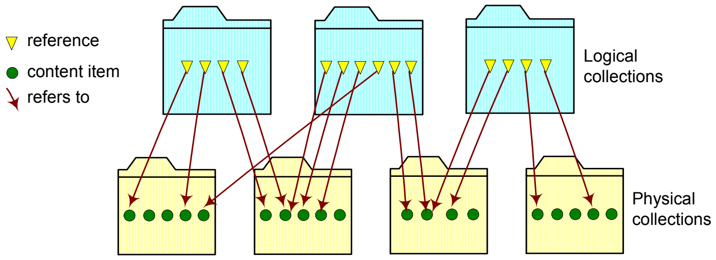

Fig. 3. Physical and logical collections

BRICKS Collections are described by metadata schemas and can be searched like any other DL item. The Application Program Interface (API) of BRICKS Collection Manager is the single access point for all the operations (local and remote) involving the DL contents (objects and metadata). Figure 4 shows the main architecture of the Collection Manager and his relations with the other BRICKS components.

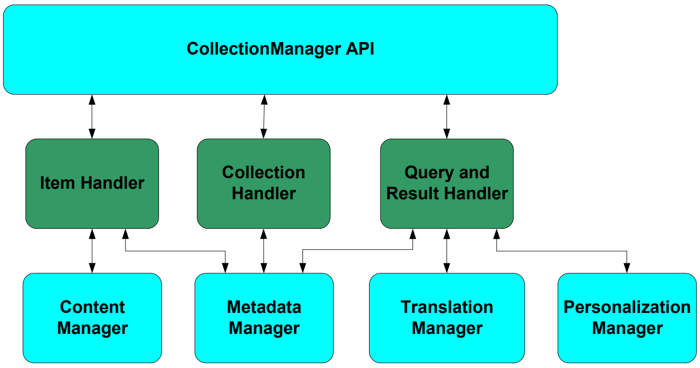

Fig. 4. Collection Manager architecture

The rectangle on the top in figure 4 represents the CM API that can be invoked by users applications via web services. The CM is composed by a set of Java classes (the boxes labelled *Handler* in figure 4) that act as bridges toward the other BRICKS components, hiding the components communications details (e.g. web service invocation, service composition, etc). The *Collection Handler* class manages collections metadata, it interact with the BRICKS *Metadata Manager*[15] component to create/update/delete a metadata repository[16] for DL items metadata records[17]. The *Item Handler* class manages DL items (digital object and metadata), it interact with the BRICKS *Metadata Manager* and *Content Manager* components to insert/remove/update digital objects and metadata, using a transaction like mechanism. The *Query and Result Handler* (described in details in the next section) manages simple and advanced queries on collections and/or items metadata, it interact with the BRICKS *Metadata Manager* to retrieve metadata records, with the BRICKS *Translation Manager* for cross-language query translation and with the BRICKS *Personalization Manager* to return records according to the user preferences.

5.1 Query Handler

The BRICKS Query Handler acts as a sort of metasearch engine [9], retrieving and combining information from multiple sources [12] and identifying useful information objects from returned results. It supports two main kinds of queries: queries searching for a list of terms, in the style of Web search engines, and structured queries where metadata schemas properties can be used to build boolean expressions. As stated in paragraph 3.1, in BRICKS each DL Item has one or more associated metadata records, conform to different schemas. Queries are executed over metadata records describing DL Items[18]. The result set of a query consists of all the metadata records of the DL items matching the query filters.

A BRICKS query can have *restrictions*, *conditions* or both, where:

- Restrictions: is a list of BNodes or collections identifiers defining the search space, i.e. the set of resources where the query will be executed (e.g. search into *Painter Collection* of ISTI BNode).
- Conditions: is a boolean expression defining the filter. Boolean condition may be expressed in terms of some schemas (Advanced Query, Ontology Query) or may be a sequence of terms with wildcards (Simple Query).

BRICKS supports the following kind of queries:

- **Simple Queries:** The language for simple query conditions is that offered by the IR system *Lucene*: it allows to express sequence of unconstrained terms. The BRICKS simple query supports cross-language retrieval. In a cross-language retrieval the condition terms are transformed in order to search for DL Items metadata written in the languages preferred by the user.

[15] based on JENA framework (http://jena.sourceforge.net/)

[16] repository is a generic name for a container of metadata records in the Metadata Manager terminology, in this case correspond to the CM notion of Collection.

[17] A Bricks Collection is in turn a DL item so has metadata records describing it.

[18] content based queries on DL object are not currently implemented.

- **Advanced Queries**. In these queries, conditions are expressions whose terms have the syntax: "*field_name operator value*" (simple conditions), where operator is one of the usual comparison operator. Each simple condition can refer to a different metadata schema (e.g. `DC.creator="Bob" AND XY.date>01.01.2000,` will return all metadata records of DL items described at least by DC and XY metadata schemas whose fields values match the filter). Note that conditions in advanced queries can also refer to the Collections metadata schema, this means either that collections can be searched and that collections metadata fields can be used as filters for finding DL Items. Advanced queries are translated into SPARQL[19] expressions and executed by the underlying Jena RDF query processor.

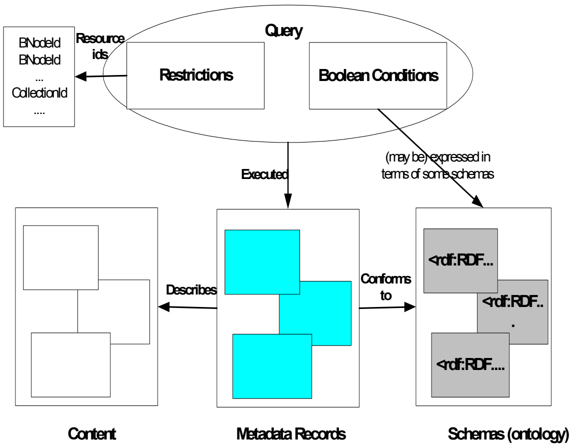

Fig. 5. Evaluating a BRICKS query

In order to make the current BRICKS implementation independent from any specific platform for query evaluation, the query service is made available through an object-based interface.

5.2 Query Processing

The goal of the BRICKS query processing is to define the search space for the query and to build a set of access plan. One of the most important processing step is the query optimisation. In the following we will describe the *source selection*, used by the query optimizer to limit the data transfer. The source selection algorithm is based on the routing indices, described in the next section.

5.2.1 Routing Indices

There are two indices that contain information about BNodes at different granularities: collections and property values.

[19] http://www.w3.org/TR/rdf-sparql-query/

- *property value pairs index* contains attribute value pairs of DL items metadata records stored in the BRICKS data space. This is used to evaluate the query conditions.
- *collection index* containing metadata of collections stored in BRICKS. This is similar to the catalog in Relational DBMS and is used to evaluate the query restrictions.

Due to the different type, amount and use of information stored in them, we have chosen two different ways to distribute indices. Property value index is stored in a Distributed Hash Table (DHT). For each property value pair a special key is computed and each BRICKS peer uses a routing table to find whom to send a message to store or retrieve a given key. The Distributed XMLStorage component is used to distribute the collection index. Metadata are stored as XML files and each BNode uses XPATH queries to retrieve a set of collection based on metadata fields conditions.

5.2.2 Source Selection

A source selection algorithm is used by the query optimizer, in order to limit the space to be searched. The source selection does the following:

1. the Boolean Condition part of the query is transformed in an equivalent DNF expression. The transformation process may require an exponential number of steps, but the number of simple conditions is not expected to be high. The *property value index* is then used to find, for every conjunctive term, the set of collections having metadata records matching it. The join of all obtained collection is the *preliminary candidate* collection space for the query.
2. if the query restriction part is not empty, the intersection of the user supplied collection space (could be a superset of the obtained one) and the one obtained in the previous phase is the *candidate* collection space for the query.
3. if the query restriction part is empty the *preliminary candidate* collection space is equal to the *candidate* collection space for the query.
4. the *collection index* is then used to restrict the *candidate* collection space to the effective collection space, by removing empty collections and collections belonging to BNodes not currently connected to the BRICKS network.
5. the access plan is then built by grouping collections by BNode.

5.3 Query execution

Basically, the access plan defines the set of Metadata Manager services that must perform the query and the order in which they will be called. To improve efficiency we have adopted an asynchronous algorithm for query execution: when a query is executed by a client application, a token identifying the query is returned. The identifier will be used by the client application to fetch metadata records in a RBMS cursor like way. Figure 6 shows a simplified sequence diagram describing query execution in BRICKS. After returning the query identifier, the BRICKS search component queries in parallel all the Metadata Managers in the access plan using a pool of threads. Every metadata manager executes the query, using local indexes, and returns the result. Results are collected by the coordinator and stored in a cache area memory for the client application.

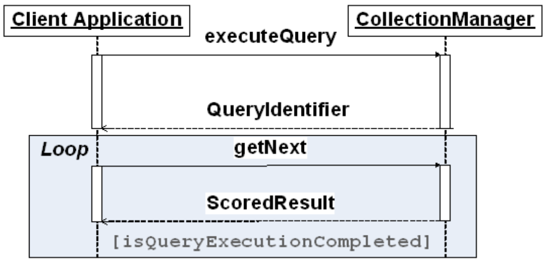

Fig. 6. Application-Collection Manager Interaction

6 Conclusions

The main features of the BRICKS DLMS have been illustrated, focusing on the Collection Manager, which is the central component of the system. BRICKS DLMS has been used to develop a certain number of applications in the area of Cultural Heritage by our project partners. A set of BNodes has been installed and are currently used by BRICKS partners (including small museums and libraries). People interested in BRICKS DLMS software can download source and/or binary code at http://foundation.bricksfactory.org/. Future works includes query optimization improvement and content based search (similarity search).

7 Acknowledgements

The BRICKS Project is partially funded by the European Commission as an Integrated Project of the 6th Framework Programme, within the framework of the specific research and technological development programme "Integrating and Strengthening the European Research Area (2002-2006)", under contract number 507457. Thanks are due to our colleagues Francesco Furfari, Maria Teresa Paratore, Federico Tonioni and Loredana Versienti who significantly contributed to the design and implementation of the Collection Manager component.

References

1. Marcos André Gonçalves, Edward A. Fox, Layne T. Watson, and Neill A. Kipp (2004). Stream, Structures, Spaces, Scenarios, Societies (5S): A Formal Model for Digital Library ACM TOIS, vol. 22, No 2, pp. 270-312.
2. William Y. Arms (2000) Digital Libraries, the MIT Press 2000.
3. Ian H. Witten, David Bainbridge (2003)“How to Build a Digital Library” Morgan Kaufmann Publishers, ISBN 1-55860-790-0

4. K. Aberer. (2001) P-Grid: A self-organizing access structure for P2P information systems. Proceedings of the Sixth International Conference on Cooperative Information Systems (CoopIS).
5. K. Aberer, M. Hauswirth, M. Punceva, R. Schmidt (2002). Improving Data Access in P2P Systems. IEEE Internet Computing, Jan/Feb 2002, pp. 58-67.
6. Borgman, C. L. (2000). From Gutenberg to the Global Information Infrastructure. Cambridge, MA: The MIT Press.
7. Lesk, M. (2004). Understanding Digital Libraries (Second ed.). San Francisco, CA: Morgan Kaufman Publishers.
8. Chowdhury, G. G., Chowdhury, S. (2003). Introduction to Digital Libraries. London: Facet.
9. W. Meng, C. Yu, and K.-L. Liu (2002). Building Efficient and Effective Metasearch Engines, ACM Computing Surveys, vol. 34, no. 1, pp. 48-89.
10. Leonardo Candela, Donatella Castelli, Pasquale Pagano, Constantino Thanos, Yannis Ioannidis, Georgia Koutrika, Seamus Ross, Hans-Jörg Schek, Heiko Schuldt. Setting the Foundations of Digital Libraries. The DELOS Manifesto. D-Lib Magazine, March/April 2007, Volume 13 Number 3/4.
11. Fragkiskos Pentaris and Yannis Ioannidis. Query optimization in distributed networks of autonomous database systems. ACM Transactions on Database Systems. Vol. 31, No 2, June 2006. Pages: 537 – 583.
12. Amit P. Sheth and James A. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. ACM Comput. Surv., 22(3):183–236, 1990.
13. Donatella Castelli, Pasquale Pagano. A System for Building Expandable Digital Libraries. Proceedings of the 3rd ACM/IEEE-CS JCDL '03
14. Robert Tansley, Mick Bass, David Stuve, Margret Branschofsky, Daniel Chudnov, Greg McClellan, MacKenzie Smith. The DSpace Institutional Digital Repository System: Current Functionality.
15. Smith, MacKenzie, Barton, Mary, Bass, Mick, Branschofsky, Margret, MacClellan, Greg, Stuve, David, Tansley, Robert and Walker, Julie H: DSpace: An Open Source Dynamic Digital Repository. D-Lib Magazine 9, 1 (January 2003).
16. Staples, Thornton and Wayland, Ross: Virginia Dons FEDORA: A Prototype for a Digital Object Repository. Dlib Magazine 6, 7/8 (July/August 2000).
17. Candela, Leonardo - Simi, Manuele - Pagano, Pasquale - Castelli, Donatella. DILIGENT: A DL Infrastructure for Supporting Joint Research. Proc. 2nd IEEE-CS International Symposium Global Data Interoperability- Challanges and Technologies. 2005. pp. 56-59.
18. Schek, H.-J. Schuldt, H. DelosDLMS: global prototype development. DELOS Research Activities 2006 C. Thanos (editor) pp. 19-20.