Design and Development of the Data Model of a Distributed DLS Architecture for Archive Metadata

Nicola Ferro and Gianmaria Silvello

Department of Information Engineering, University of Padua, Italy
{ferro, silvello}@dei.unipd.it

Abstract. In this work we present the architecture of a *Digital Library System* (DLS) that enables the preservation, management and sharing of archival descriptive metadata in a distributed environment. Furthermore, we describe in detail the design and development of the data model envisioned for this DLS. The result is a flexible and scalable architecture ables to deal with the complexities of archival metadata.

1 Introduction

The role of *Digital Library Systems (DLSs)* in collecting, managing, sharing and preserving our cultural heritage is increasingly prominent in several contexts. DLSs have become the fundamental tool for pursuing interoperability between different cultural organizations such as libraries, archives and museums. Collecting and managing the resources of these organizations is fundamental for providing a wide, distributed and open access to our cultural heritage. One of the most important issue to be addressed is the interoperability between different cultural organizations which may differ in their organizations, policies and data management procedures.

In this work we consider the archives which are a complex and challenging cultural organizations. When archives are considered, the most relevant resource type that must be taken into account is metadata. In the archival context metadata are called archival descriptive metadata and express the archival descriptions; these are the foremost digital resources preserved by the archives. Indeed, most archival documents are not available in digital form, but they are described and represented by metadata. In the work we have been carrying out we have underlined that DLS technologies need to be revisited to be well-suited and successfully applied to the management of archival metadata and digital objects [3].

In [4] we presented a distributed DLS architecture to address the problem of sharing archival metadata between different archives spread across a geographic region. In particular, we considered the Italian Veneto Region archives which promote a related project called *Sistema Informativo Archivistico Regionale (SIAR)*. The main goal of the SIAR project is to develop a DLS for sharing archive metadata spread across the territory. Archive metadata are geographically distributed across the Veneto Region and they are preserved in several local archives; the SIAR objective is to develop a DLS able to provide advanced services on regional archive metadata [1].

The DLS architecture designed in the SIAR project is divided into three basic layers: the data exchange infrastructure described in [2,4], the metadata management layer and the user interfaces layer. The main goal of this work is to present and describe the data model of the SIAR system realized throughout the metadata management layer of the DLS architecture.

Section 2 reports the background of SIAR principles which are worthwhile for understanding the design and development of the data model. Section 3 presents the design of the metadata management layer of the SIAR DLS and Section 4 reports some final remarks.

2 SIAR architecture and the design choices

The SIAR architecture is based on three main layers built one upon the other: the transportation layer, the metadata management layer and the user interfaces layer. A schematic view of this three-level conceptual architecture is shown in Fig. 1.

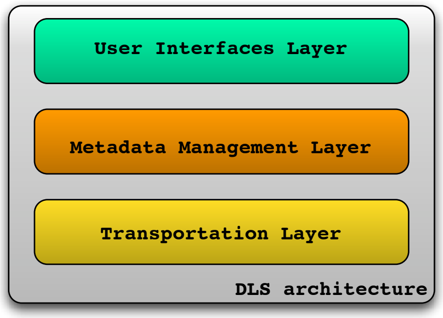

Fig. 1. The three main layers of the conceptual SIAR DLS architecture

The transportation layer is represented by the data exchange infrastructure. The main role of this infrastructure is to permit metadata exchange between the local archives spread across the territory. Basically, it is composed of the *Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH)* [11] protocol that is a standard de-facto for metadata exchange in distributed environments. OAI-PMH is based on the distinction between Data Provider and Service Provider which, respectively, offer metadata and harvest metadata to provide services. Data Providers are the components that make metadata available to the Service Providers that harvest metadata. Each Data Provider manages its own metadata and it is independent and autonomous from the outside information systems. The Service Provider role is to harvests metadata by the different Data Providers and to performs advanced services on these harvested metadata. In our case study, the Veneto Region is the Service Provider which gives advanced services such as data and public access to the harvested metadata and the archive keepers act as Data Providers because they supply archive metadata. OAI-PMH requires *Dublin Core (DC)* metadata format as minumum requirement; DC, a tiny and lightweight metadata format, is becoming the preponderant means for the exchange of information in a wide distributed environment.

The intermediate layer is the metadata management level that has to manage archival metadata retaining their full informational power; when archival metadata are considered several issues must be taken into account. Indeed, archival metadata reflect the structure of the archive which is strongly hierarchical and related to the organization preserving them; an archive organizes its documents describing them from the general units to the specific units in a tree structure. Every document is linked with the other documents of the same hierarchy and, at the same time, with the environment in which they were created and preserved. In order to describe the archival documents correctly, the metadata have to retain the relationships between the documents themselves and with the production and conservation environments; in other words, they have to maintain the hierarchical structure and the context of archival documents.

Typically, archival metadata are described by metadata format encoded in big and complex *eXtensible Markup Language (XML)*[1] files able to retain hierarchy and context information; a relevant example is the *Encoded Archival Description (EAD)* metadata format [7] which is the standard defined by the Society of American Archivists in cooperation with The Library of Congress. EAD encourages archivists to use collective and multilevel description, and because of its flexible structure and broad applicability, it has been embraced by many repositories [6]. The use of EAD is widespread in the United States of America and also in the European Union; for instance the "Nationaal Archief"[2] in the Netherlands preserves a big collection of EAD metadata in Dutch or the "Archives Napoléon"[3] is based on EAD metadata in French. Unfortunately, EAD allows for several degrees of freedom in tagging practice, which may turn out to be problematic in the automatic processing of EAD files. Moreover, EAD files are heavy and difficult-to-move. Moreover, the EAD file cannot be accessed with a variable granularity; indeed, to access a specific archival unit it is necessary to visit all the archival tree [9], starting from the general fonds that usually is the tree root and going down the node path until the required unit is found. It has been underlined [8] that the EAD metadata standard is not well-suited for use in a distributed and dynamic environment like the SIAR is. Indeed, the SIAR system does not rely on a specific metadata format that can turn out

[1] http://www.w3.org/XML/

[2] http://www.nationaalarchief.nl/

[3] http://www.archivesnationales.culture.gouv.fr/chan/chan/archives_napoleon-averti.htm

to be problematic in a specific domain, but it leaves the choice to the archives participating the system.

At the same time, we have to consider that not every metadata format is well-suited to deal with archival descriptive metadata. For instance, the Dublin Core metadata format is not enough informative to be used in a stand-alone way in the archival context. For these reasons we have defined a data model based on nested sets which permits to handle full informative archival metadata using small and flexible metadata formats like the Dublin Core [5]. In particular, the XML metadata files must be organized in a proper way that enables the preservation of hierarchy and context information. By means of the use of sets we can retain the full informative power of archival metadata; the sets are nested one inside another to create a hierarchy that reflects the archival organization. We can say that the archival information is brought by the documents content and by the archive structure; in the SIAR architecture the content is brought by the metadata and the structure is retained by the nested sets organization. Metadata and sets are the two basic logical entities constituting the metadata management layer of the SIAR DLS architecture.

The third level of the SIAR DLS architecture is the presentation layer constituted by the user interfaces. The system presents two main interfaces: the first is a general-purpose interface dedicated to a generic user-type such as archivists, historical researchers, public administrations or private organizations that will use the advanced services available in the SIAR DLS; the second is dedicated to specialized users who can use this interface to add, remove or update archival metadata.

3 SIAR Metadata Management Layer

The SIAR metadata management layer is the central component of the DLS architecture. Throughout this level it is possible to manage, preserve, retrieve and share full expressive archival metadata.

In Fig. 2 we can see a sketch of the 3-layers architecture with a zoom on the metadata management level composed by: the database, the data logic and the application logic. The data logic is realized by a component called *datastore*, instead the application logic is composed by the *service manager* and the *web component manager*.

We can clearly see the four main entities of the SIAR DLS: metadata, set, user and group; the main function of the database and the datastore is to create, read and update the various instances of these entities. Furthermore, they supply the data to the application logic. The service manager realizes the various services of the SIAR DLS such as the representation of metadata, the reconstruction of the set organization and the OAI-PMH data and service providers. The web component manager implements those methods that permit the interaction of the services with the web services exploiting the SIAR DLS. Currently we have designed a web service that realizes a user interface interacting with the SIAR DLS; in Fig. 2 it is represented as the presentation logic.

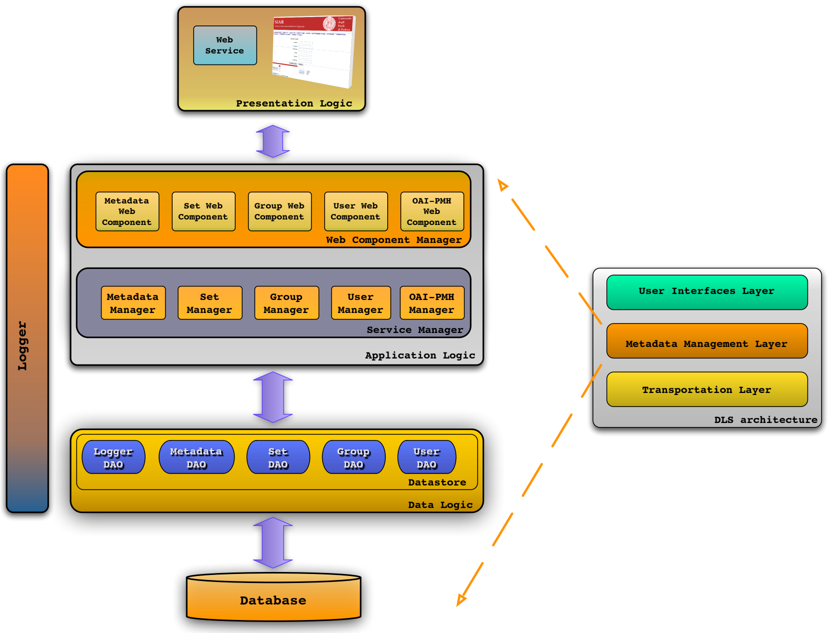

Fig. 2. Composition of the SIAR metadata management layer

The second component presented in the Fig. 2 is the logger. The logger keeps track of all the activities of the system at all the levels; indeed, it is the only component transversal to the whole metadata management layer. The logger registers all the accesses of the users and all the operations done by the system such as the creation of metadata, sets, users or groups, the access and any update of the entities or the exceptions risen and handled by the system.

The elements composing the metadata management layer are described in the following three subsections: subsection 3.1 presents the database conceptual schema in order to describe the entities treated by the SIAR DLS. Subsection 3.2 describes the structure and the components of the datastore constituting the data logic of the layer and subsection 3.3 explains the role of the service manager and the web component manager composing the application logic.

3.1 SIAR Database Design

The conceptual design of the database reflects the world that the SIAR DLS represents. In Fig. 3 we can see the conceptual schema of the database.

The conceptual schema is composed of four main entities: metadata, set, user and group. The metadata entity is defined by five attributes: *id*, this is the unique identifier of a metadata; the identifier is assigned automatically by the system, calculating an hash function of the metadata content. *body* containing

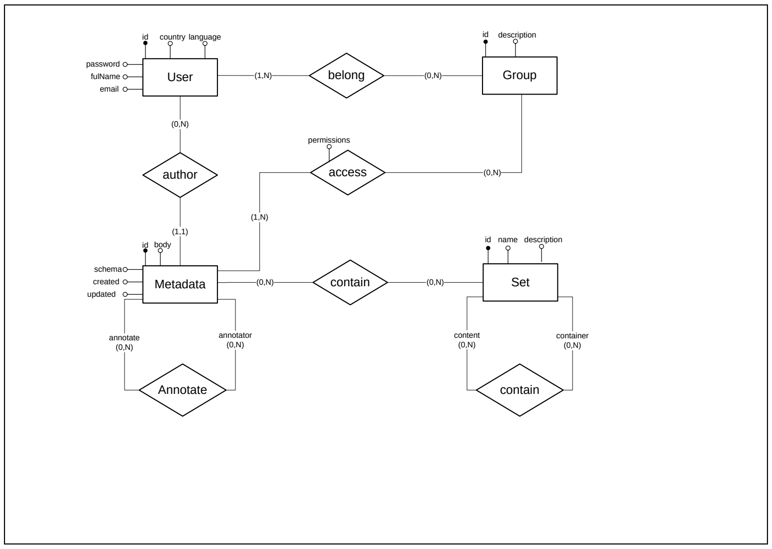

Fig. 3. SIAR DLS Conceptual Database Schema

the metadata content encoded in XML; the *body* is defined in the database as a native XML data type. Every metadata format encoded in XML can be stored in the database and the *schema* attribute reports the metadata format; throughout this attribute we are able to individuate the metadata format and to parse the XML file in a correct way. *Created* which is the attribute storing the time stamp in which the metadata was created in the database and *updated* which stores the time stamp in which a metadata could have been updated.

The recursive relationship called *annotate* indicates if a metadata annotates (*annotator*, the metadata is the annotator of other metadata) some other metadata or if it is annotated (*annotation*) by other metadata. Thanks to this relationship we can have notes on metadata expressed as metadata themselves, we can retain the metadata history preserving all the versions of metadata that have been modified and we can retain information about the original repository of the metadata if they have been harvested by means of the OAI-PMH protocol. A metadatum can annotate many other metadata, for instance in the *history case* an old metadatum is the annotator of its updated version and the *repository case* where the metadatum describing a repository is the annotator of all the metadata coming from that repository. Furthermore, annotate recursive relationship is useful to represents nested metadata. Metadata are the most important entities in the SIAR system and they establish relationships with all the other entities. The *author* relationship specifies that a metadatum must be created by a user also if it has been harvested via OAI-PMH protocol. The *belong* relationship indicates that a metadata can belong to a specific set or to many sets defined in the system; a metadatum can also belong to no set at all. The *access* relationship indicates that each metadatum must have at least one group with read or write permissions on it (e.g. a metadatum has to be read at least by the group to which the creator user belongs) and can have many groups with permissions on it.

The *set* entity is defined by three attributes: *id* which is the unique identifier of a set, *name* which is a mandatory attribute indicating the name of the set and an optional attribute called *description* which can contain a free text description of the set. There is a recursive relationship called *contain* which indicates if a set is contained (it is a subset) by other sets or if it contains (it is a superset) other sets or both. Throughout this recursive relationship we express the possibility of creating a nested set organization that expresses the hierarchical structure of an archive. The structure of this entity is complaint with the structure of an OAI-set defined by the OAI-PMH protocol; indeed, it defines an OAI-set as a native feature composed by three main components: *setspec* which is the identifier of the OAI-set, the OAI-set name and the description which may contain free text or a piece of XML. This connects in a straightforward way the SIAR with OAI-PMH and permits to exploit an important feature as the selective harvesting which is the procedure that permits the harvesting only of metadata owned by a specified OAI-set [10].

The *user* entity is defined by six attributes: *id* which is the unique identifier of a user that can be seen as the username to access the system, *password* the password chosen by the user to access the system, *fullname* contains the full name of the registered user, *email* is the e-mail address of the user, *country* and *lang* indicate the origin of the user and are useful for setting up multilingual services. The *group* entity is defined by an *id* and a *description* of the group. The group and the user entities are related by the *contain* relationships that establish to which groups a user belongs. A user is not required to insert a metadatum to participate in the system but the permissions to create, read and update metadata are related to the group, thus users have to belong to at least one group to operate on metadata.

Fig. 3 does not show the entity called *log* that stores all the events handled by the logger of the SIAR system.

3.2 Data Logic

The data logic component of the metadata management layer is constituted by the SIAR datastore that defines all the methods that the application logic may call on the data logic of the SIAR system. The SIAR datastore is independent from any particular *DataBase Management System (DBMS)* and is composed of several components called *Data Access Objects* (DAOs). The DAOs are used to abstract and encapsulate all access to the database; the DAO manages the connection with the database to obtain and store data. Essentially, the DAOs act as adapters between the components and the database. In Fig. 2 we can see that the data logic is composed by five DAOs: the metadata DAO, the set DAO, the user DAO, the group DAO and the logger DAO. Every single DAO defines all of the methods that have to be provided for managing the corresponding entity in the database.

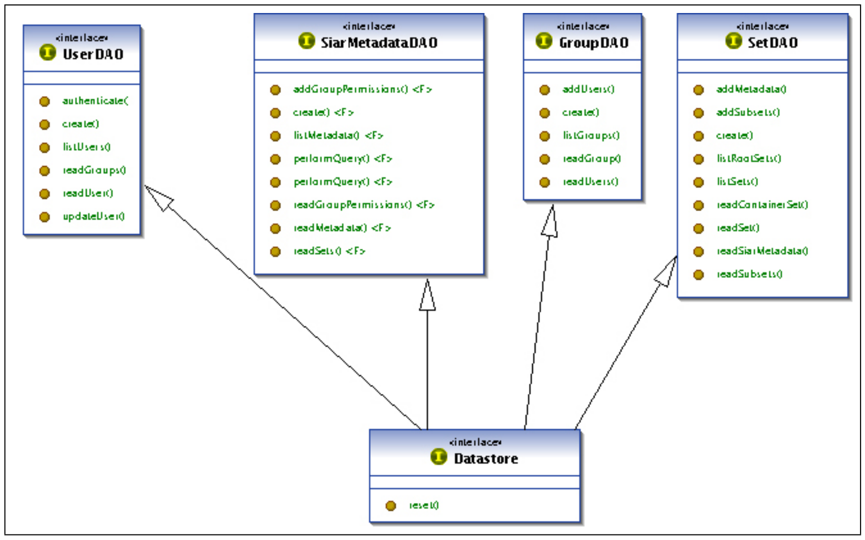

Fig. 4. DLS Data Logic: The SIAR Datastore

In Fig. 4 we can see the methods defined in the five DAOs of the SIAR datastore. For instance the Metadata DAO implements methods for creating or reading metadata, for adding a metadatum to a set, for listing all the metadata belonging to a set, for adding read and write group permission or for reading all the set to which a specific metadata belongs. In the same way Set DAO implements methods for creating a set and adding and reading metadata from the sets. Furthermore, Set DAO defines several methods for obtaining the hierarchy of supersets of a set or the list of subsets.

The other DAOs define the operations on the other SIAR entities.

3.3 Application Logic

The application logic is constituted by two components: the service manager and the web component manager. The service manager defines all the functionalities provided by the SIAR system; instead the web components manager defines all the methods that provide support for the web services implementation and elaborate external requests.

In Fig. 5 we can see how the service manager is composed: there is a service for each DAO in the datastore and an OAI-PMH service that implements the protocol functionalities. For instance the *user management service* provides the *authentication service* that permits the users authentication or the *reset password service* that allows a user to change his password.

The OAI-PMH service realizes the data and service provider features exploiting the other SIAR DLS services. The data provider component of this service answers the requests of external service providers; for instance it creates lists of

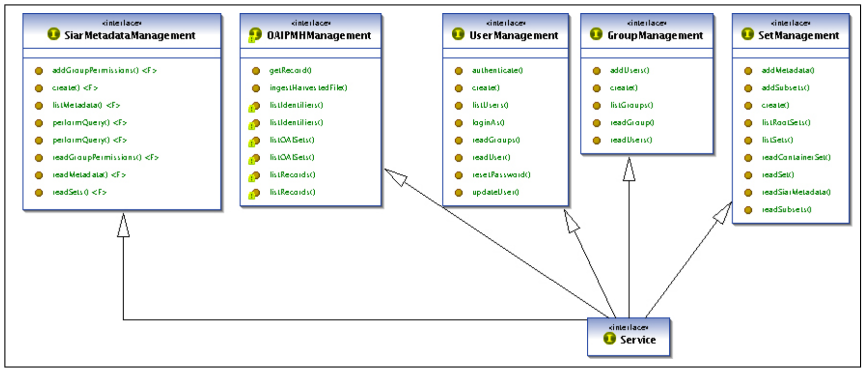

Fig. 5. DLS Application Logic: The SIAR Service Manager

metadata encoded in XML files or returns the set organization of the system formatted as an OAI-PMH response. The service provider component enables the harvesting of metadata from other repositories and their storage in the SIAR DLS. In the same way external services can also be developed and added to the SIAR.

4 Final Remarks

We presented the SIAR DLS architecture that enables the sharing of metadata between several archives in a flexible and scalable way. We explained the SIAR design choices, describing in detail the data model of the DLS represented by the metadata management layer.

Future work will concern the continuation of the development of the presentation layer composed by the user interfaces.

Acknowledgments

The study is partially supported by the TELplus Targeted Project for digital libraries, as part of the eContentplus Program of the European Commission (Contract ECP-2006-DILI-510003)[4]. The work of Gianmaria Silvello was partially supported by a grant from the Italian Veneto Region.

References

1. M. Agosti, G. Bonfiglio-Dosio, N. Ferro, and G. Silvello. *Metodologie e percorsi interdisciplinari per la ideazione di un Sistema Informativo Archivistico.* Memoria dell'Accademia Galileiana in Scienze, Lettere ed Arti in Padova, Ente di Alta Cultura (D.P.R. 27/10/49 n.1005), 2008.

[4] http://www.theeuropeanlibrary.org/telplus/

2. M. Agosti, N. Ferro, and G. Silvello. An Architecture for Sharing Metadata among Geographically Distributed Archives. In C. Thanos, F. Borri, and L. Candela, editors, *DELOS Conference*, volume 4877 of *Lecture Notes in Computer Science*, pages 56–65. Springer, Heidelberg, Germany, 2007.
3. M. Agosti, N. Ferro, and G. Silvello. Proposta metodologica e architetturale per la gestione distribuita e condivisa di collezioni di documenti digitali. *Archivi*, 2(2):49–73, December 2007.
4. N. Ferro and G. Silvello. A Distributed Digital Library System Architecture for Archive Metadata. In M. Agosti, F. Esposito, and C. Thanos, editors, *Post-proceedings of the Forth Italian Research Conference on Digital Library Systems (IRCDL 2008)*, pages 99–104. ISTI-CNR at Gruppo ALI, Pisa, Italy, July 2008.
5. N. Ferro and G. Silvello. A Methodology for Sharing Archival Descriptive Metadata in a Distributed Environment. In B. Christensen-Dalsgaard et al., editor, *Proc. 12th European Conference on Research and Advanced Technology for Digital Libraries (ECDL 2008)*, pages 268–279. Lecture Notes in Computer Science (LNCS) 5173, Springer, Heidelberg, Germany, 2008.
6. K. Kiesling. Metadata, Metadata, Everywhere - But Where Is the Hook? *OCLC Systems & Services*, 17(2):84–88, 2001.
7. Society of American Archivists. Encoded Archival Description: Tag Library, ver. 2002. Society of American Archivists, 2003.
8. C. J. Prom. Reengineering Archival Access Through the OAI Protocols. *Library Hi Tech*, 21(2):199–209, 2003.
9. S. L. Shreeves, J. S. Kaczmarek, and T. W. Cole. Harvesting Cultural Heritage Metadata Using the OAI Protocol. *Library Hi Tech*, 21(2):159–169, 2003.
10. H. Van de Sompel, C. Lagoze, M. Nelson, and S. Warner. Implementation Guidelines for the Open Archive Initiative Protocol for Metadata Harvesting - Guidelines for Harvester Implementers. Technical report, Open Archive Initiative, p. 6, 2002.
11. H. Van de Sompel, C. Lagoze, M. Nelson, and S. Warner. The Open Archives Initiative Protocol for Metadata Harvesting (2nd ed.). Technical report, Open Archive Initiative, p. 24, 2003.