Project Number IE-2005

Project Title Querying heterogeneous data in Aquarelle project[1]

Deliverable Type Research Report

Deliverable number D4.6

Contractual date of delivery

Actual date of delivery 20 September 1997

Nature of deliverable

Authors Vassilis Christophides[†], Paola Carrara[‡], Marco Padula[‡], Michel Scholl[*], Anne-Marie Vercoustre[*]
[†]FORTH-ICS, P.O.Box 1385 GR 711 10 Heraklion, Crete, Greece
[‡]ITIM-CNR, Via Ampere 56, I20131 Milan, Italy
[*]INRIA-VERSO, BP 105, 78153 Le Chesnay Cedex, France

Abstract The Aquarelle project aims at developing a distributed multimedia information system, offering access to information describing and documenting European cultural heritage such as painting, sculptures, historical sites and monuments, musical instruments and furniture. Cultural information, is currently provided by many cultural organizations (museums, libraries, galleries, etc.) across European borders, under various forms and structures (record, document, image, drawings or text bases etc.), using different information management systems (DBMS, IRS, KB) and platforms. One of the main objectives of the Aquarelle project is to provide a uniform access to these heterogeneous collections of data.

In this paper we present the limitations of the current Aquarelle architecture to explore the structure of available data sources and we suggest some guidelines and solutions for the future version of the Aquarelle information system. More precisely, we split our recommendations into two parts: some more realistic ones for the near future, some more ambitious ones requiring a deeper evolution of the current architecture. Furthermore, we review recent projects on heterogeneous database querying and integration that go beyond traditional IRS keyword or full text based search. This state of the art illustrates possible long range functionalities of Aquarelle. Two demonstrations[2] have been done in conjunction with these recommendations to illustrate some current trends in query interoperability against heterogeneous data sources.

Keywords Querying Heterogeneous data, POQL, metadata, structured data, semi-structured data, cultural information

[1]Part of this work was done while the first author was at INRIA.
[2]http://africa.itim.mi.cnr.it and http://cosmos.inria.fr:8080/poql.html

Contents

Executive Summary

The Aquarelle project aims at developing a distributed multimedia information system, offering access to information describing and documenting European cultural heritage such as painting, sculptures, historical sites and monuments, musical instruments and furniture. Cultural information, is currently provided by many cultural organizations (museums, libraries, galleries, etc.) across European borders, under various forms and structures (record, document, image, drawings or text bases etc.), using different information management systems (DBMS, IRS, KB) and platforms. One of the main objectives of the Aquarelle project is to provide a uniform access to these heterogeneous collections of data.

In this paper we present the limitations of the current Aquarelle architecture to explore the structure of available data sources and we suggest some guidelines and solutions for the future version of the Aquarelle information system. The limitations of the current Aquarelle architecture result mostly of providing a flat view of information at any level: user interface, Query language, Access server, wrapping of data sources for using the Z39.50 protocol.

We split our recommendations into two parts: some more realistic ones for the near future, some more ambitious ones requiring a deeper evolution of the current architecture. Short term improvement consists in providing more information to the end user to help him in selecting the data sources, or issuing native queries to a specific repository. This is related to a better integration and use of metadata in the system.

Furthermore, we review recent projects on heterogeneous database querying and integration that go beyond traditional IRS keyword or full text based search. This state of the art illustrates possible long range functionalities of Aquarelle. We propose three main directions: Querying heterogeneous information using incomplete structure (POQL), a multidatabase approach using a Common Database model powerful enough to represent overlapping semantics, and the coupling of a KBS-information system with the Access server and the user interface. We also suggested to add geo-referenced navigation and querying to the system to better support the geographical data which play an important role in cultural heritage applications.

Two demonstrations[3] have been done in conjunction with these recommendations to illustrate some current trends in query interoperability against heterogeneous data sources.

[3] http://africa.itim.mi.cnr.it and http://cosmos.inria.fr:8080/poql.html

1 Introduction

Sharing cultural information across European borders such as paintings, sculptures, historical sites and monuments, musical instruments and furniture, is the main challenge of the Aquarelle project. The Web is more and more used by museums and cultural organizations for broadcasting cultural contents (see for instance the sites of Louvre[4] or Orsay[5]), as well as for exchanging cultural information at an international level. Current Web technology provides the necessary standardization and global communication infrastructure, but existing tools are somehow limited to manage information in a professional framework (poor data structures, dangling hyper-links, *ad-hoc* search engines, etc.). The Aquarelle project, has joined the Web space for interacting with available cultural information and intends to provide a reliable and effective access to heterogeneous data sources within an open federation of organizations for mutual benefit.

Museum curators, urban planners, commercial publishers and researchers should be able to collect information relevant to their needs or interests notwithstanding the information location and organisation. In addition, each author of a given information component should be able to link directly a part of his/her own creation to another information asset created and updated by another author. Linking and commenting relevant pieces of information belonging to different sources will bring much more than a simple access to existing information: it will add value to the information contents itself. The overall Aquarelle architecture is designed to relieve users from the cumbersome manual task of maintaining cross references as well as to support the high precision required in referencing and retrieval.

In order to develop such hypermedia networks of multimedia documents, Aquarelle relies on two main sources of cultural information: existing primary material, called *archive data* such as records, drawings, maps or text bases provided by the different cultural organizations (museums, galleries, etc.) and secondary material, referred to as *folder data*, in the form of SGML documents, describing, commenting and referring archive data, as well as adding new raw information. Folders, in the sense of Aquarelle, are considered as containers gathering a structured collection of specific information elements (archive data), which can be semantically linked together (intra or inter folder references). The following are examples of folders : a) folders on architectural sites, monuments or objects, etc. such as those actually handled by the "Inventaire General" of the French Ministry of Culture[6] b) folders prepared by a museum curator as supporting documentation during the preparation of an exhibition c) folders created by a commercial publisher to edit a city guide.

In order to create a folder, users can start to retrieve, via queries on archive and folder servers, cultural heritage information related to their study or research interest. Furthermore, they can browse through the retrieved folder's structure and hyperlinks to identify particular objects of interest. Then, the user can insert into the new folder, references to the relevant objects, via an SGML editor. Users will "cycle" between editing, retrieval and browsing until they achieve a satisfactory product. *Therefore, queries on data available in archive and folder servers play a central role within the Aquarelle information discovery system.*

Since there has been little coordination between the cultural organizations in setting archive

[4] http://mistral.culture.fr/louvre/
[5] http://www.cnac-gp.fr/sommaire.html
[6] http://aquarelle.inria.fr/Inventaire

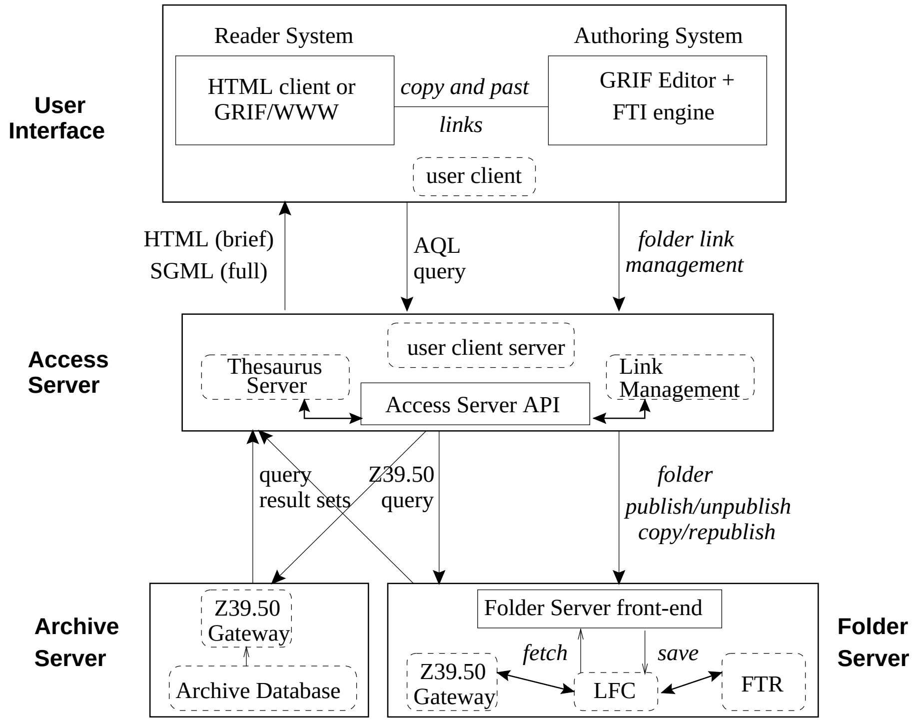

Figure 1: The Architecture of the Aquarelle Information System

servers, existing primary material is currently available under various forms and structures (records, texts, images, drawings, databases, etc.), and it is managed by quite different platforms and systems : Data Base Management Systems (DBMS), Information Retrieval Systems (IRS), Knowledge Base Systems (KBS). In addition, describing, commenting and referring primary material within folders should be performed with respect to the folder SGML structure defined by a Document Type Definition (DTD). Aquarelle is designed to support an open number of DTDs according to the user needs. In order to bootstrap the first evaluation phase, the project starts with the CIMIs DTD called Cultural Heritage Information On-Line (CHIO) [7]) and that of the French Ministry of Culture, "Classeur d'Inventaire"[8] (CI). Other DTDs such as Text Encoding Initiative [9] (TEI) or HTML[10] should also be used in a later phase of the project.

In this context, *there is no uniform or fixed in advance data schema* and the cultural information structure ranges from strongly-structured (in record-oriented relational bases or graph-oriented object bases), to semi-structured where the structure is looser or irregular or implicit (in SGML or bibliographic bases) until unstructured raw data (images or drawings).

Querying these heterogeneous data sources, in a transparent way for end-users, is the central issue of the Aquarelle project.

The rest of this paper is organized as follows: section 2 describes the Aquarelle architecture.

[7]http://www.cimi.org/cimi/CHIO.html
[8]http://aquarelle.inria.fr/Inventaire
[9]http://etext.virginia.edu/TEI.html
[10]http://www.w3.org/hypertext/WWW/MarkUp/html3/CoverPage.html

Its current limits are discussed in section 3. For enriching the access to heterogeneous sources, section 4 suggests short-term solutions which have the advantage of respecting the current architecture. Related work in the field of federated databases, multidatabases and heterogeneous databases is presented in section 5. More ambitious perspectives for Aquarelle are then studied in section 6.

2 The Aquarelle Architecture

The architecture of the Aquarelle Information System is shown in figure 1. Its central module is the *Access Server*, which mediates between archive databases, folder servers, on the one hand (using the Z39.50 protocol) and user clients (user client server) on the other hand [DSWM96]. It is also interfaced with a thesaurus server [DF97]. It is responsible for users authorization, resource discovery, Z39.50 query handling, and one-to-one connection with the data servers. It also supports metadata handling for the various data sources and guarantees referential integrity of folder's hyperlinks.

The Aquarelle *User Interface* [Mar97], consists of the reader system and the authoring system. The former allows Aquarelle end-users to access available data archives and published folders. It supports retrieval through the *Aquarelle Query Language* (AQL) as well as browsing through the folder hyperlinks. The reader system is implemented with a Web client solution: either usual HTML clients (Netscape, for example), or full SGML clients (using for example GRIF/WWW) configurated as helper applications. The authoring system utilizing the GRIF SGML editor is an environment suitable for creating and modifying structured folders according to the selected Aquarelle DTDs (CIMI, CI, etc.). During folder editing, the user can create hyperlinks to archive and folder data (using adequate URLs) and add content descriptors (semantic information) to folders under the discipline of the DTD. The latter is supported by on-line queries on the thesaurus server. Finally, keyword extraction can be performed using Full Text Indexing (FTI) integrated in the SGML editor.

The main components of the *Folder Server* [DBF96] are the Local Folder Catalogue (LFC), and the Full Text Retrieval System (FTR) which is not yet available in the current version. A dedicated front-end coordinates the cooperation of the LFC and the FTR with the rest of the Aquarelle architecture. More precisely, it maintains communication with the Access Server, and manages storage and access to folder instances. The LFC is based on the Semantic Index System (SIS) to manage SGML documents (publishing, unpublishing, etc.) as trees characterized by attributes [CDF97] and provides a navigation interface as well as a state-full query language. The latter is used by the Z39.50/SIS Gateway to evaluate the queries and return the relevant folder or folder parts via the Access Server to the Aquarelle end-users.

Finally, *Archive Servers* are configurated to handle Z39.50 queries on top of exisiting cultural databases (developed with relational DBMS as well as IRS such as ORACLE, SYBASE, INDEX+, BASISplus, Mistral, etc.) using appropriate gateways [SLS97].

3 Limitations of the current Aquarelle Architecture: a *flat* view of information

Most Web search engines available today (e.g., Altavista[11], Lycos[12], Excite[13], Yahoo[14], etc.) are based on keyword search, mostly provided by text indexing. In order to improve retrieval quality in terms of recall/precision, Aquarelle information discovery system is designed to provide in conjunction with traditional keyword or full-text based search (i.e., non-controlled terms) a thesaurus based search using controlled vocabulary. The emphasis is on the multi-lingual aspects of cultural information management and on the automatic translation of queries from users native language to the corresponding thesauri (AAT, MERIMEE, RCHME, etc.) used to index the various data sources.

However, keyword or concept retrieval is more suitable for querying collections of unstructured or semi-structured documents, than structured data sources. Indeed, the current Aquarelle architecture is strongly inspired from a flat IRS view of information, based on record-oriented bibliographic descriptions using a set of so-called Access Points (AP). Aquarelle APs are derived from the CIMI-CHIO efforts to provide a uniform Z39.50 based access interface to heterogeneous cultural information available on the Internet.

Aquarelle APs allow to retrieve parts of a cultural documentation (in archive or folder servers) related to a particular concept (i.e., "This is about that person ... event", etc) or having a particular meaning for museum specialists (i.e., "this is about that artist ... historical context", etc.). *Access points correspond to relations, classes or attributes* (or even a combination of them) in the database jargon and they are used to give a scope for Z39.50 queries. Sometimes these access points are "hardwired" in the document structure, as in the case of the CIMI DTD, sometimes they must be mapped onto the real data structures (archive DBMS schemata and folder SGML DTDs such as CI).

The advantages of this "minimal" approach to heterogeneous data querying are obvious:

- "Legacy" cultural databases (i.e., archives) are not altered within the Aquarelle network: Access Points are not used to describe the various cultural objects but just as a support to map local data structures into a "common access model".
- Semantic discrepancies among heterogeneous data sources are roughly captured: Access Points are used as a "minimal access model" leaving apart any semantic subtelty encountered in the integrated databases.
- Mediation between the Access Server and heterogeneous data servers is simplified: query translation, routing to archive or folder servers as well as fusion of answers are straightforward.
- Integration of new information sources is trivial: it requires only a simple registation of a new source to the Access Server by giving its network address and the domain describing its contents.

[11] http://www.altavista.digital.com/
[12] http://www.lycos.com/
[13] http://www.excite.com/
[14] http://search.yahoo.com/

However, in this context, most of the existing structure and semantics richness of the archives and folders is lost for querying.

There is no translation foreseen between richly structured archives and folders on the one hand and the Access Points view in the Access Server, on the other hand. Instead the semantics of the data server is artificially translated into a flat set of access points. Note that structured information exists not only in folder servers (SGML documents) but also in several archive servers (DBMS based), although it might be *unknown*, or *hidden*, or to be *discovered*. In particular, the data source structure is extremely useful:

1. for facilitating query refinement and improving precision, compared to keyword or full-text based search;
2. for addressing more easily fine grain chunks of information compared to hypertext navigation;
3. for enabling sophisticated data integration from various data sources;

No exploitation of the existing structure in folder servers as well as in many archive servers is the main and most significant limitation of the current Aquarelle architecture.

This limitation concerns the communication with the data sources and the information model used for mediation by the Access Server as well as the Aquarelle Query Language and the User Interface.

Querying and integrating heterogeneous sources of data is a difficult and challenging issue which could bring a much better service to the end-user. Different approaches have been proposed in the database literature (see section 5) to cope with the problem of query interoperability among heterogeneous databases and many systems have been developed with motivations similar to that of the Aquarelle project. Providing advanced querying and integration facilities of heterogeneous cultural information is an ambitious task that comes with an extra complexity in the Aquarelle architecture, in particular at the Access Server level (see section 6). However, there exist some short term solutions which do not require a drastic change in the current architecture (see section 4).

The flat minimal view of cultural information justifies the current choices for: (i) Z39.50 as a communication protocol with the data sources; (ii) the Aquarelle Query Language; (iii) the relatively poor mediation services provided by the Access server and (iv) the rather simple user interface. If we were to take into account the data sources (archive and folder servers) richness in structure, the above components of the system present some serious drawbacks we describe now. We also argue that the integration of two functionalities in the same module (i.e. the Access server), namely query mediation and link management is not really justified.

3.1 Drawbacks of the Z39.50 protocol to handle structured data

One of the basic choices of the Aquarelle project is to follow standard technology at the data and communication levels. In this way, access by existing systems and future service providers can be facilitated, and longer life-time of the data created in the Aquarelle environment is guaranteed.

One of the key standards used in Aquarelle is the Z39.50 protocol [Z3996] for an open and reliable access among heterogeneous data sources.

The goal of Z39.50 (version 3 or later) is to minimize the amount of "knowledge" about local database structures required by a client in order to retrieve data from it. In the Z39.50 jargon, the Access Server is acting as a client (stated as origin) towards the Archive or Folder Servers (stated as servers or targets). Then the used Z39.50 Profile i.e., the Access Points, can be viewed as a "Universal Relation" shared by clients and servers, which can be queried and which must be mapped onto each real archive or folder data structure. More precisely, on the target side, we need to define an "attribute set" corresponding to the APs, to unambiguously characterize a query from a client to a server (i.e., data servers input). On the origin side, we need to define an abstract response record structure in terms of a common "set of elements" both the target and origin communicate through (i.e., data servers output).

For example, suppose that a database server (target) has records with fields known locally as *Title*, *Actor*, and *Event_Date*. For intersystem retrieval, the target may map these fields to the (semantically) corresponding Access Points in the predefined Z39.50 Profile, as for example, the attributes *title*, *publisher*, and *date*. In addition, the target may define a fixed number of fields to be returned in the result sets, characterizing the retrieved records for all queries. A client then could receive a Z39.50 response record consisting for example, of three elements: *recordid*, *title*, *creator*; since the client and server both share an understanding of the Z39.50 profile, the client knows the intended semantics of these three pieces of data in the record.

It becomes clear that the underlying information model of the protocol is "record"-oriented (inspired from bibliographic text bases). Furthermore, the concept of "record" or document in the traditional IRS view is intrinsically "flat", compared to the view of records or objects in DBMS (or of document fragments in modern SGML based IRS). Finally, retrieval (using attribute sets) of relevant records and presentation of results are two distinct actions in IRS, while in DBMS the user must specify the required fields (*select clause* in SQL) together with the filtering condition (*where clause* in SQL). It is worth also noticing that, unlike DBMS, the structure of returned records is fixed (i.e., query independent) as it is (pre-)defined in the Z39.50 Profile.

These *semantics differences in data representation and access are the main obstacles to issue structured queries and retrieve complex data via the Z39.50 protocol.* The query capabilities of the Z39.50 wrapped data sources are limited to a) boolean filters and b) projection on fixed attributes; they do not allow associative access (joins).

3.2 Poorly structured view of Information by the Aquarelle Query Language

The Aquarelle Query Language (AQL) is based on a flat view of Information corresponding to the adopted Z39.50 Access Points.

AQL allows to specify abstract queries consisting of a set of query elements that are combined by Boolean operators (and, or, and-not). AQL is not used directly by the users. In some way the user interface converts the user inputs in AQL queries using the available Access Points and the supported retrieval operators (=, truncations, etc.). For example, in order to find paintings and sculptures by Alberto Giacometti containing "Man" somewhere in their title, users should issue the following query using the GUI:

access$*personal_name*="Alberto Giacometti" and access$*title*, trunc$both=Man[15]

Each query element consists of a term or of one or more qualifiers (separated by commas) followed by an equal sign and a term. These qualifiers are translated into Z39.50 attributes provided in the Aquarelle profile. This is a one-to-one mapping. The qualifier names have two parts (separated by a dollar sign) namely a qualifier type (i.e., the different types of Z39.50 attributes for access points, retrieval operations, used authorities, etc.) and the qualifier value (e.g. access$*personal_name*).

If now users want to retrieve only paintings of Giacometti, the following condition must be added to refine the initial query: access$*type_or_classification*=paintings. Note that the fact that fine art archives may have separate database relations for sculptures and paintings, is not captured by AQL.

Let us consider now a more complex example. Usually a number of actions (creation, acquisition, donation, possible modifications, etc.) are associated to cultural objects, where an action is characterized by at least one actor and an event. An event is determined by a time and a place. Then a user may want to retrieve the work of Alberto Giacometti made when he joined the Surrealists in 1930. This query can be expressed in AQL as follows:

access$*personal_name*="Alberto Giacometti" and access$*event*=creation and access$*date*="1930:1940"

The user will receive as a response to this query a number of records satisfying the search conditions. But nothing guarantees that the retrieved cultural objects have indeed been created by Alberto Giacometti in the given period. In fact Alberto Giacometti may have donated the returned panting or sculpture to a museum, or even restored it. Any logical association between actions, events, persons, dates and places have been lost.

It is important to note that *the structure of data is extremely useful to improve query precision* but AQL is too simple to express structure complexity.

Furthermore, *data integration from multiple sources is impossible.* With the current Aquarelle architecture there is no way to get specific paintings (or sculptures) of Alberto Giacometti together with the associated documentation available in different archive or folder servers. The flat view of information by the Access Server does not allow any useful logical interconnection of the data sources in the Aquarelle network and any construction, as an output to the user, of a new document from the document selected from a single source or from several sources by an AQL query. The only thing we can do is select a fixed subset of the fields of a selected document (projection in the database jargon).

3.3 Mixture of link management and query mediation

The current Access Server conception (see figure 1) mixes mediation (for query handling) and data warehousing (for folder link management). In order to provide a reliable access to Aquarelle data pointed by hyper-links in folders, a normalization of folder and archive data IDs and link management (validation of link destinations) is also performed by the Access Server. In this

[15] "both" stands for left and right truncation.

context, the same folder contents are analyzed at least four times : (i) at the SGML editor level, for consistency with the DTD, (ii) by the Full Text Indexing System (for keyword extraction), (iii) by the Access Server (for link extraction and management) and (iv) by the Folder Server (for loading in the SGML repository). Although, extremely useful to guarantee data integrity (compared with the current chaotic Web situation) the implementation of link management at the access server level raises 4 difficulties :

- Referential integrity for fine grain links is not completely achieved: modifications on specific folder fragments (during republishing) cannot be captured by the Access Server that should withdraw the links pointing to the updated data.
- Transaction management for folders manipulation is complex: a cooperation of loosely coupled and distributed modules is required to overcome system or network failures.
- Link management is not scalable: a) keeping centralized bases of links in the Access Server (at least for each country) will be quite counterproductive as the system is growing up, b) there is no master module responsible for the management of links in the Aquarelle network c) the whole distribution aspects of the Aquarelle Access Server are not yet specified.
- Sophisticated queries on folder's SGML structure/hyperlinks (e.g., incoming or outgoing links) and contents (e.g., concepts or keywords) is not possible since link information is stored outside the folder servers: only the data sources are responsible for Z39.50 queries.

We believe that a more elegant solution should be to leave the management of folder contents and links at the folder server level (with a minimal cooperation of the Access Server as a control module) while the Access Server should focus more on its mediator role for queries.

3.4 Poor mediation power of the Access server for queries

Besides, due to the adoption of the Z39.50 view of information, Access Server mediation is poor compared to proposed mediation models in the literature (see section 5). It is limited (i) to the translation of Aquarelle Query Language (AQL) into Z39.50 queries for broadcasting to the data servers, (ii) the multi-lingual expansion of the access points values in queries and (iii) a simple fusion of disassociated results returned by the data sources.

Given that the integrated data sources (archives and folder servers) contain not only heterogeneous cultural information but also provides different levels of query capabilities (whether they are IRS or DBMS based), the Access Server follows a "lowest common denominator" approach to deal with all sources. Indeed, the whole architecture is adapted to the more primitive data sources (i.e., IRS based archives). Analysis of user queries, decomposition into a set of queries tailored to specific query capabilities of data sources, and optimal post-processing of the results such as data integration from partial answers, duplicate elimination, are few examples of the mediation services that should be supported.

Of course more knowledge of the information structure at the Access Server and at the User Interface implies a more complex integration process but has for a consequence richer querying capabilities than those actually provided by AQL.

3.5 Missing features in the Folder server

The current folder server design has two limitations :

First, the full text retrieval capability has not yet been integrated : the focus when designing folder servers has been on representing, managing and querying folders with complex structure and links toward archive data or other folders, which features are missing in current IRS. DBMS allow to fulfill such requirements, but have no facility for efficient full text retrieval. The current trends are to couple the DBMS to an efficient keyword based text retrieval engine. The coupling between a folder database on the one hand and a full text retrieval engine is far from being solved. Several issues have to be considered : (i) which part of the folder should be indexed? (ii) should the index refer to the folder(s) satisfying a keyword or to a fragment within the folder? (iii) how to deal with queries mixing structure and text, (iv) how to optimize such queries. Only items (i) and (ii) have been addressed in the context of Aquarelle. The current folder server design must be enhanced in order to allow for efficient full text retrieval.

The second limitation concerns the mapping of the folder structure to APs : there is currently no mapping between APs and folders. Instead folder profiles have been defined as Dublin core like metadata and are mapped to APs.

3.6 Limited User Interface

The current Aquarelle user interface[16] provides a minimal set of services allowing a simple and uniform access to heterogeneous data.

In order to access the Aquarelle services, the user is first invited to provide his login identifier and password. This administration information as well as various user preferences (language, automatic translations, time-outs, etc.) are handled in the so-called user profile. Then, the user can start the retrieval of cultural information by selecting target sets and formulating (AQL) queries. The target set is a list of data servers and domain names such as Architecture, Painting, etc. The formulated queries will be in the sequel sent to all the data servers related to the selected domains and the other servers named explicitly in the target set.

As stated above, an Access Point view of information allows for the simplest access to heterogeneous data servers. The only current effort toward letting more information perspire out of the data source to the end user is through the use of data domains. The Access Server is then in charge to provide all the information about the registered domains and servers.

However, such an end user interface has restricted functionalities. More semantics about the data sources and more knowledge at the conceptual level should be available at the end user.

We shall discuss in the sequel two types of solutions to improve the Aquarelle services in terms of access to heterogeneous sources. In section 4 we suggest short term solutions which have the advantage of respecting the current Aquarelle architectural choices. In section 6 we look at long range perspectives which allow for a more powerful interface and query language for accessing cultural heritage heterogeneous data sources. Of course such perspectives imply a more profound change in the current Aquarelle architecture.

[16] http://falconet.inria.fr:8282/

4 Possible Extensions to Aquarelle Architecture

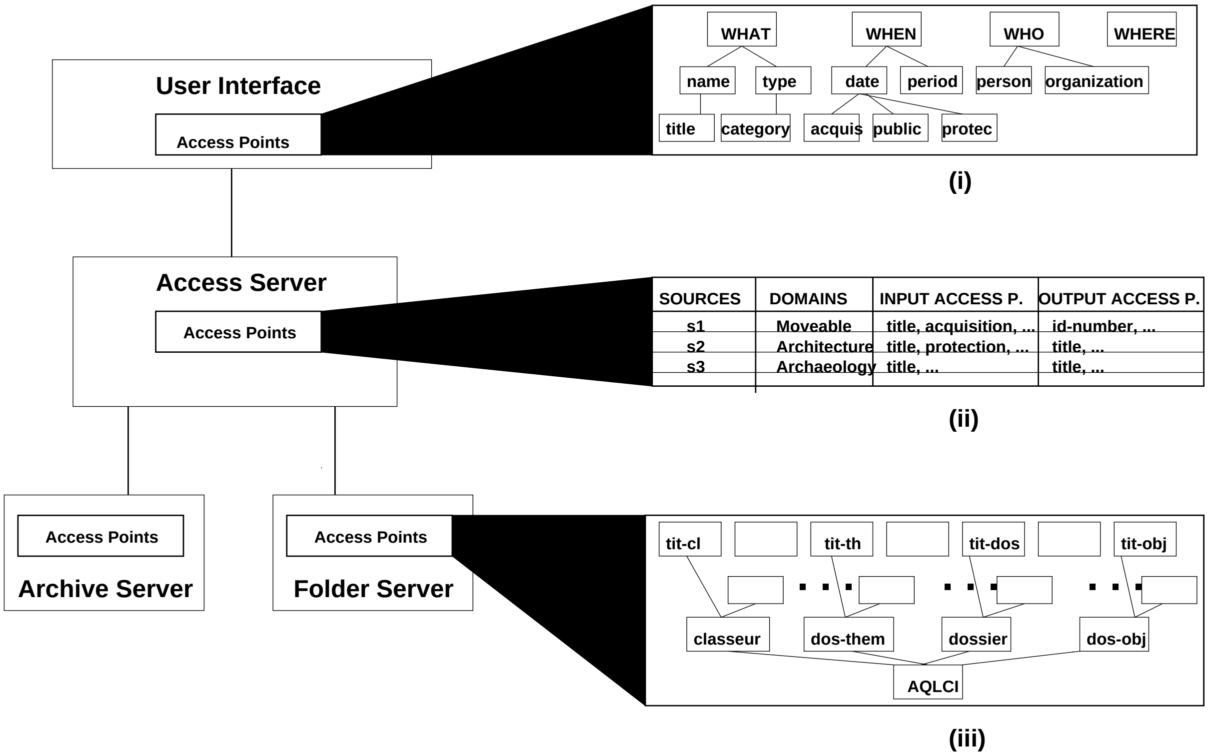

Figure 2: Short Term Extensions of Aquarelle Architecture

Sticking to the Access Point flat view of information we suggest in this section (i) to improve the user interface by more knowledge at the conceptual level (ii) to provide a better integration of metadata in the Access Server, (iii) to allow some simple structure querying to specific data sources, (iv) to use the Z39.50 Explain facility, and (v) to improve the text retrieval facility of the folder servers. Figure 2 illustrates some of these short range solutions which are compatible with the current Aquarelle architecture.

4.1 Helping the user in his/her query formulation

More semantics about the data sources should be available to the end user. In particular, the user should have the possibility of browsing the metadata related to data sources. More knowledge about the metadata of data sources not only allows the user for a better refinement of his/her needs (and therefore improve precision), but also allows him/her to express structured based queries which actually are not supported by the system. This perspective may be exploited both at the user interface level to present the meta information of specific databases and at the Access Server level via a common database schema giving a global view of the available information.

In order to enrich the current user interface, access points based conceptual modeling should be explored.

The discussion on the Aquarelle forum and the suggestion of a hierarchical classification (specialization/generalization) of Access Points (see figure 2 case i), referred to as 4W (i.e., WHAT, WHEN, WHO, WHERE) for architectural heritage, is only one simple example of conceptual modeling. This classification should help the user in his/her choice of the relevant Access Points and values to choose for his AQL queries.

Currently the semantics of the adopted Access Points for Aquarelle, mixes the description of the objects with the cultural objects themselves. This is true more for archive data than for folders (which by definition are a documentation related to the objects). For instance, the AP *title* may correspond to the title of a related documentation (i.e., description record) or the title of a cultural object (i.e., painting, sculpture, etc.) referenced within the document. A clear distinction of the different semantics of access to heterogeneous data will better orient users during retrieval and reduce noise in the returned information.

An automatic generation of AQL queries according to user selections in the above hierarchy of APs is one of the possible extensions of the GUI. For this a similar approach to the thesaurus access based to AP values is sufficient. A complementary extension of the GUI is of course to present only the APs which are related to servers selected by the user.

Finally, it must be stressed that the user interface should depend not only on the data sources available but also on the user perspective. This implies more sophisticated user interface customization than simple linguistic translations currently provided, and various user helps adapted to various levels and natures of knowledge of the data servers on which the user wants to restrict his/her attention. Clearly, there is a need for a variety of user interface paradigms depending on the user expertise: naive user or curators, urban and regional planners, publishers and researchers, etc. We already mentioned that users could query the various sources based on the data sources structure. Users could also have a multi-step way of querying, refining the query according to intermediate answers.

4.2 Better Integration of metadata in the Access Server

There has recently been a significant reflexion in the Aquarelle project (see WP6 and spring Pisa meeting) toward *metadata*. Although not initially part of our task, we also contributed a little to this reflexion [Ver97].

To our opinion metadata can be classified into three categories : (i) information that can be derived directly from contents such as structure of the data, (ii) information that can be derived by system tools such as size, format, etc., (iii) information that cannot be derived directly from data content and structure. For instance, a Subject description of folders is situated in a more abstract level than its Title.

The discussion about "metadata for digital objects" was started by information communities trying to provide uniform semantics descriptions for accessing unstructured data (images or drawing bases) or semi-structured data (HTML pages where the SGML tags serve only for presentation reasons). In all cases, it is currently understood that schema (in the Database jargon) is part of the metadata. In addition, the query capabilities of the data sources (i.e., support of boolean or SQL queries) as well as constraints of the source's contents (e.g., data only for a specific time, place, etc.) is also part of the metadata. In the current design, APs represent metadata from all sources, i.e. structure as well as information that cannot directly be derived from the data contents. However at the access server level, there is no way to decide whether a given source has a valid instance of a given AP.

There are two ways of handling metadata : (i) either it is up to the data servers to manage there metadata and export them to the Access Server or directly to the GUI (ii) or it is the responsability of the Access Server to integrate and maintain the metadata of the data sources.

In the former case, the data sources must provide the metadata in the format expected by the Access server or the GUI. In this context the use of the Z39.50 Explain facility can be considered (see subsection 4.4). In the latter case, which corresponds to the current design, an enhancement of the registration process of the data sources at the Access Server must be provided, i.e. the access server should at least know which source has actual values for a given AP (see figure 2 case ii). Then metadata can be used by the Access Server for filtering the most appropriate data sources to answer a user query.

But it is also a first important means for increasing the knowledge about data sources at the user level as a help for the user to issue queries. Metadata should be part of a more sophisticate knowledge base system integrating not only domains and/or access points but also shared semantics between the different data sources.

Integrating metadata at the Access Server level can also be seen as a first step toward the definition of a *Common Data Model* (CDM) (see multidatabase/federated systems in section 5). Local databases with different schemas (or structures) are associated into a synthetic logical schema which provides a common reference for query formulation [Fur96]. In such an architecture, due to the autonomy of the local systems, it is always possible to query the archives separately. But for more advanced searches it is necessary to establish where the schemas have semantic overlaps. This can be done by means of the description of the archive domain as a set of attributes: D = (n1,t1),(n2,t2,),...,(nn,tn,) where (ni, ti) is a pair (field name, field type). A field name is the attribute name of the database jargon while the field type is not the field structure (as in strongly typed databases) but an informal description of the field content. For a formal discussion of these concepts and of the transparent access to data sources with overlapping domains [17], see [CCG+97].

4.3 Structure-based Queries to specific data sources

We have argued that the knowledge of the actual data structure at the user level might help the user in query formulation (without modifying AQL).

For the time being we assume that some users might be interested in accessing specific archive or folder servers whose structure they want to discover, or they are aware of (see figure 2 case iii). This is *not a uniform access to heterogeneous data sources*, but on the contrary an enrichement of the user interface for the user that wants a narrower and richer access to a more restricted range of servers.

Experience from existing Web interfaces (see for instance the sites of Mediaculture[18], or JO-

[17] This has been implemented and is demonstrated in http://africa.itim.mi.cnr.it.

[18] http://aquarelle.inria.fr/Inventaire/

CONDA and MERIMEE[19]) assisting users to formulate structured queries tailored to specific cultural DBMS is very useful. Furthermore, a direct exploration of specific database schemata by dedicated Web/DBMS interfaces (e.g., query designers for commercial DBMS[20]) allows users for dynamic navigation/querying at both data and structure levels.

Once the structure is known, users might issue more complex queries to be understood and interpreted by the DBMS/IRS based archive or folder servers. For instance, in a folder server with SGML documents describing architectural objects (i.e., CI DTD) one may issue queries like: "Find buildings photographs whose caption contains Place" or "Find the buildings whose all bibliographic references concern Cognac Region". As we have seen in section 3.2 such queries *cannot easily or not at all be expressed with the current flat view of information by AQL.*

Since we do not want here to modify the current Aquarelle architecture, these queries will be expressed using the native query language of the target system, and they will be sent via the Z39.50 protocol as uninterpreted "strings" (i.e., Z39.50 type 0 queries). The same stands also for returned query results.

The above idea might be improved by the use of an extension of Z39.50 to handle SQL queries, called Z39.50/SQL+ as for instance Version 4[21] [CF96] (see section 5). It should be looked at as well as the modifications it implies in the Access server. While type 0 queries require an one-to-one connection with the data sources, the second solution (Z39.50/SQL+) requires a deeper modification the current Aquarelle architecture to prepare a correct and processable query, tailored to the specific limitations of the data servers, either IRS or DBMS based.

4.4 Exploring metadata through Z39.50

Z39.50 allows also some facilities to explore metadata. In particular the Explain facility manages a database with information such as name, description, protection, cost of the databases which are available for search, or more specific data such as access points details, schemas, record syntax, diagnostic, etc. Explain categories, stored as tables have to be specified in the search syntax. Categories of main interest for our goal are:

- TargetInfo for information about the target;
- DatabaseInfo for searching a specific database managed by the target;
- SchemaInfo describes the schema of a database. Databases with different structures could be manages by a same target;
- TermListInfo is a category of the database Explain which includes all descriptive information about the database access points (which are similar to the previously introduced idea of attribute) which could be usefully exploited to implement the domain.

A Z39.50 search operation can then specify the request for all databases managed by the target which receives the query : for example, a request such as ExplainCategory='DatabaseInfo' AND

[19]http://www.culture.fr/culture/docum.htm
[20]http://cosmos.inria.fr:8080/WIRE_Demo/main.html
[21]http://www.dstc.edu.au/DDU/projects/ZINC/

Availability='yes' gives as a result the references of all the databases managed by the target. The expression ExplainCategory='TermListInfo' AND DatabaseName='<database name>' gives as a result all the term lists associated with the databases managed by the target.

The analysis of the profile for Digital Collections [Z3996] and the proposal Z39.50/SQL+ [CF96] shows the possiblity of storing into the Explain database also metadata of Web or relational archives.

4.5 Efficient text retrieval by folder servers

One simple extension to the current functionalities of the folder server is toward efficent full text retrieval. The idea here is to integrate the Aquarelle existing folder indexation with the folder server. This implies several tasks : (i) integrate the index with the folder server or provide an API for doing it, (ii) efficient retrieval of folders when the query involves an access to the index : this implies first to use at the query levelthe same linguistic treatments as for folder indexing as well as optimal query evaluation in the presence of a full text index.

5 State of the Art on distributed DBMS

The objective of this section is to review past and current trends in distributed database management systems.

Providing integrated access to multiple, distributed, heterogeneous, autonomous databases and other information sources is a topic that has been studied in the database research community for well over a decade. It is our belief that the following should help in better understanding the challenges in accessing distributed heterogeneous cultural information.

We first present the Multidatabase/Federated approaches and describe briefly some related projects. Although it is now a dating technology from which systems have evolved, it provides a framework to understand the main issues in database integration and especially the idea of providing integrated or federated schemata.

New-generation systems are interested in integrating not only data stored in databases but data that reside in other sources, possibly with not structured or with implicit structure, such as the Web or Information Retrieval Systems (IRS). In this context the focus is more on how to retrieve information from the various data sources and to integrate the results rather than providing global transaction services (eg. updating). As a result, the need for a global schema, or even federated schemas is relaxed: The emerging mediation services embed only the knowledge that is necessary for processing specific sources of information.

5.1 Multidatabases/Federated DBMS Approach

The concept of multidatabase emerged in the early 80's, because of both the limitations of conventional distributed databases, and the wish of reusing old but crucial applications in new technological environments. By multidatabase, one usually refers to systems able to provide a user transparent interaction with multiple sources of information which are distributed (as

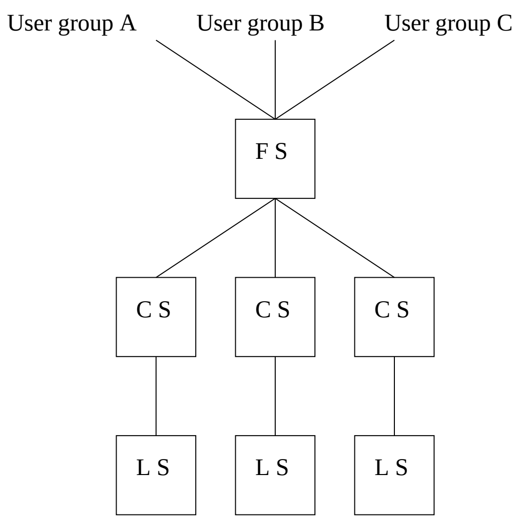

Figure 3: The Global Schema approach

each component may reside on a different node of a network), heterogeneous (as for data model, query language, data identifiers, etc.), and autonomous (as each component maintains a degree of control and may preexist to the global system and evolve after its connection to the global system).

The reader may find deeper information in overview papers such as [BHP92], [LMR90], [SL90], [PBE95], [Fah94]. The latter paper recognizes that the various proposed architectures do not present distinct boundaries but seem to form a continuum of *alternatives from tightly coupled to loosely coupled systems.*

The common feature to all multidatabase architectures is the existence of a Canonical or Common Data Model (CDM) to reduce the complexity of the problem of mapping data and commands between the different data models and languages of the component sources.

In the following figures, local schema (LS) stands for the conceptual schema of a component database system. The local schema is translated into a component schema (CS) which represents the same information satisfying the CDM (due to this translation, the CDM should be semantically as rich as the underlying component data models). The component schemas may be integrated in one or more integrated or federated schemas (FS) which are also expressed in the CDM. In the Global Schema architecture (see figure 3) there is only one, unique integrated schema which is maintained by the multidatabase administrator. It is the most tightly coupled approach and it minimizes local sites autonomy. As it requires integration of all information, it provides a simple access to users, conflicts resolution, and global integrity constraints, but it is not realistic when the number of components is very large. If more integrated schemas are created and maintained by the global administrator, as in the Multiple Integrated Schema approach, each users' group has access to information integrated in one schema for its special needs. A federated schema can be accessed from different users sites. Only information that is known to be used together is integrated.

Following [HM85], the term Federated approach is appropriate if there is no global maintenance: the single user sites are in charge of integrating needed information from the component schemas

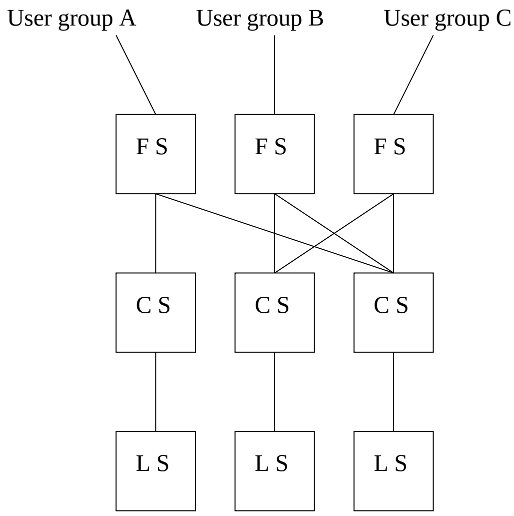

Figure 4: The Federated approach

(see figure 4). It is obvious that different user sites cannot share a federated schema.

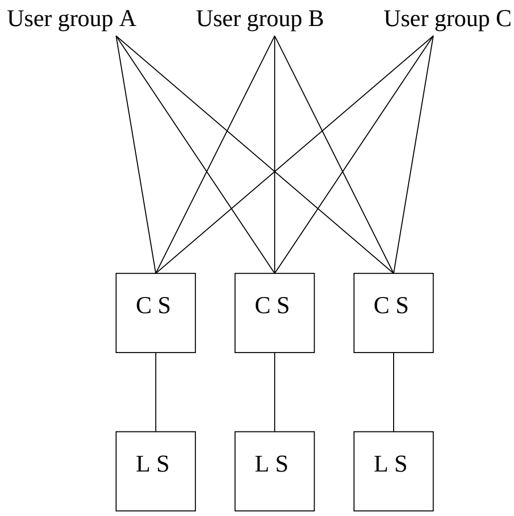

Figure 5: The Multidatabase Language approach

If no integration level is provided, users have no filtering layer which makes transparent the access to the various components: in the Multidatabase Language approach it is the language that provides constructs to manipulate, access and combine data from the multiple components (see figure 5).

It is interesting to note that heavy knowledge translation operations have to be performed in all above approaches: a global administrator, the administrator at user sites, or users themselves through language constructs have to map information of interest from the local component to the upper layers. These operations cannot be avoided and the expressiveness of the common model should be at least as rich as the richest component model.

5.1.1 The PEGASUS project

The Pegasus project[22], [PBE95], [Fah94], carried out by the Database Technology Department of Hewlett Packard Laboratories, provides access to native and external autonomous databases. Pegasus is defined as a heterogeneous information and process flow management system. External databases, which are accessible through Pegasus, but not directly controlled by it, are managed independently by external data resource management systems that can have different data models (relational, object-oriented, network, etc.), query languages, world views. They are autonomous and grant varying degrees of access privileges to Pegasus, ranging from simple retrieval of data to both retrieval and update of data and schemas. Pegasus' authors claim that external resources can be also file systems, legacy systems, Geographic Information Systems(GISs). Pegasus uses the IRIS data model as CDM and an extension of OSQL (HOSQL) as data manipulation language. Its architecture is composed of three layers (see in figure 6): an external resource is represented by its imported schema. The translation from the local schema and the importation are performed in a single step, using the view mechanism of the extended OSQL.

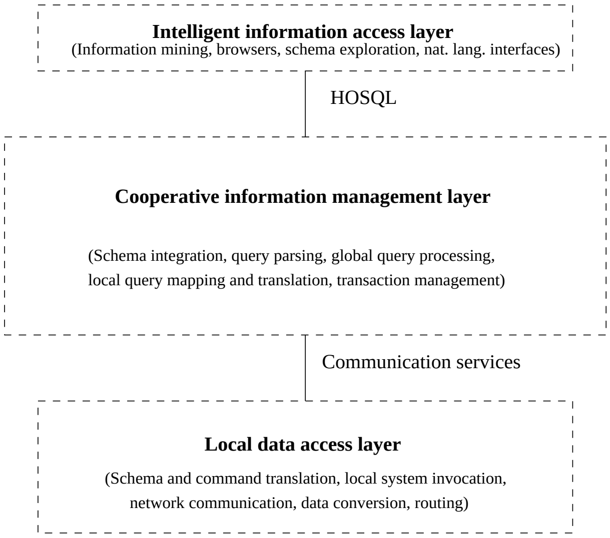

Figure 6: The Pegasus architecture

The user issues HOSQL queries against the undefined schema of Pegasus. These queries are decomposed into a set of possible parametric subqueries, each one of which refers to data residing in a single external database. In a parametric subquery some predicates include parameters whose values are received from other subqueries at evaluation time. A first Pegasus prototype was developed in 1991 and integrates two relational databases and a GIS. A second prototype (1994) integrates DB2, Informix, Oracle, and Sybase.

[22] http://www.coe.uncc.edu

5.1.2 The AMOS multidatabase

AMOS (Active Mediators Object System) [Fah94] is a research project on active databases and multidatabases of the University of Linkoping; the AMOS multidatabase architecture (see in figure 7) is a hybrid one, as it combines characteristics from the multiple integrated schema, the federated, and the multidatabase language approaches in order to minimize their disadvantages.

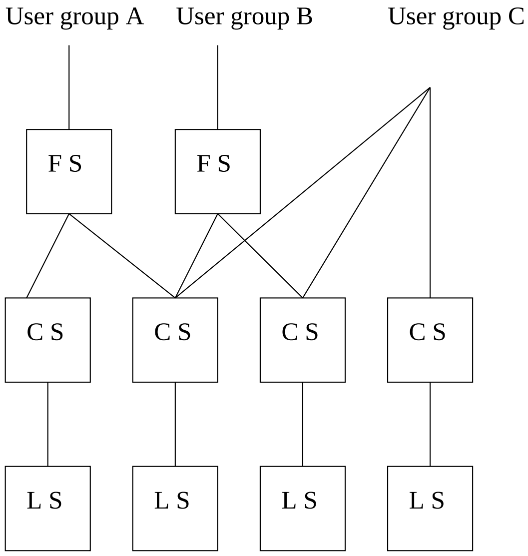

Figure 7: The AMOS architecture

Like Pegasus, AMOS's CDM is based on the IRIS object-oriented data model and uses an extension of OSQL as a data manipulation language. In AMOS, however, translation and integration of local schema are performed separately: software components called *translators* map local schemas in the corresponding component schemas (expressed in the CDM), and transform queries into calls to the underlying data sources. There is one translator for each kind of source. Users or application programs may access directly to a translator using the multidatabase language. Other software components called *integrators*, map component schemas onto federated schemas which are defined by means of the view capabilities of the multidatabase language. An integrator also decomposes a single user query into several queries against the translators and processes results to form the answer to the initial query. In AMOS each translator only needs to know the data model of one kind of data source, while a Pegasus server must understand all underlying data models.

5.1.3 The Garlic project

The Garlic project[23], [RS97] is developed by the members of the database and machine vision groups in the IBM Almaden research center in San Jose, California. Although it is not defined as a multidatabase system, it is described in this section as it presents to the users a global o-o schema. However it is described as a middleware system providing a unified schema and

[23] http://www.almaden.ibm.com

common interface of a variety of existing heterogeneous data sources. The storage systems are independent, of any type and any number. The data sources have their own data model (relational, o-o, etc.), schema, programming interface, query capabilities. Each data source (or repository) is encapsulated in a wrapper which mediates with the middleware (see in figure 8): it describes the repository's content (by a wrapper schema) and query capabilities in the system data model (based closely on the ODMG standard), and translates users' queries against the language of the repository managing system.

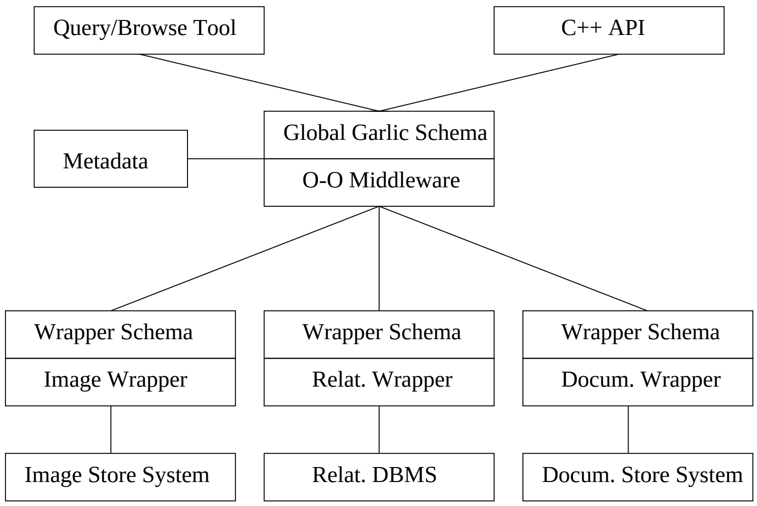

Figure 8: The Garlic architecture

The middleware query processor which takes o-o queries against the global schema, subdivides them, and redirects pieces to the right repositories (also content-base queries can be submitted on image data in repositories managed by the QBIC system[24]). The Garlic middleware gets back and glues the answers from the wrappers of the repositories. It's worth noticing that the Garlic metadata does not include information about the query processing capabilities of the repositories: it identifies portions of a query relevant to a data source and the wrapper determines how much of the work is feasible. The system can be accessed through C++ APIs or directly through an *ad hoc* user interface. The Garlic query language is an extension of SQL supporting path expression, nested collections, and methods. To date, about 10 wrappers have been developed, but authors claim that wrappers are quite easy to be written.

5.2 Information Integration Systems using Logical Views

Although Garlic and Pegasus have been introduced in the multidatabase section, they have actually evolved towards systems able to integrate other data sources than databases. The particularity of this "new-generation" information-integration systems is that, especially with the explosion of the Web, they admit the fact that an important volume of data resides outside conventional DBMS (relational or object-oriented, etc.) [Vas94]. Indeed, most of the time information is collected in documents and handled by editors, navigators, search engines, etc.

[24] http://wwwqbic.almaden.ibm.com/ qbic/qbic.html

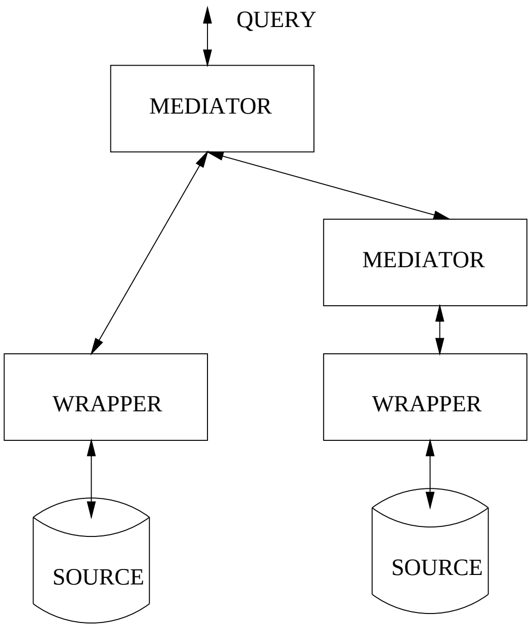

Figure 9: Common Integration Architecture

In this context, the traditional multidatabase-federated approaches are not particularly suited to access data managed by IRS systems, form-based Web interfaces, etc. A common integration architecture that currently emerges [Ull97] is shown in figure 9. Several sources are *wrapped* by software that translates between the source's local language, model, and concepts in the one hand, and the global concepts shared by some or all of the sources, on the other hand. System components, called *mediators*, obtain information from one or more components below them, which may be wrapped sources or other mediators. Mediators also provide information to components above them and to external users of the system.

A mediator embeds the knowledge that is necessary for processing a specific type of information. The key points are that a mediator does not need to understand all of the data it handles, and no person nor software component needs to have a global view of all the information handled by the system. The mediator may also process answers before forwarding them to the user, say by converting dates to a common format, or by eliminating articles that duplicate information. The second task is clearly more complex than the first one since figuring out that two articles written by different authors say "the same thing" requires real intelligence during post processing of query results.

In a sense, a mediator is a logical view of the data found in one or more sources. Data does not exist at the mediator level, but one may query the mediator as if it stored data; it is the role of the mediator to access its sources and find the answer to the query. In the sequel, we will present two alternative mediator architectures, namely the TSIMMIS and the Information Manifold projects. Related Information Integration projects found in the literature are: HERMES[25],

[25] http://www.cs.umd.edu/projects/hermes/overview/paper/index.html

SIMS[26], DISCO[27] and the Internet Softbot[28].

5.2.1 The TSIMMIS project

The Goal of TSIMMIS [29] (Stanford University) [CGMH+94] (Stanford University) is a) to develop tools that facilitate the rapid integration of heterogeneous information sources, and to ensure that the information obtained is consistent b) not to perform fully automated information integration that hides user diversity but to provide a framework and tools to assist end users and/or humans programming integration software in their information processing and integration activities.

Focuses are on providing integrated access to very diverse and dynamic information which may be unstructured or semi-structured, often having no regular schema to describe it. Information access and integration are interwinned. Integration in this environment requires more human participation. There is no global database schema. Transactions across multiple information sources usually are not provided, and each information source may support different capabilities for accessing and monitoring data.

Figure 10 shows a collection of (disk-shaped) heterogeneous information sources. Above each source is a *translator* (or *wrapper*) that logically converts the underlying data objects to a common information format, called OEM (Object Exchange Model). To do this logical translation, the translator converts queries over information in OEM into requests that the source can execute, and it converts the data returned by the source into the common OEM format.

Above the translators in figure 10 lie the *mediators* refining in some way information from one or more sources. Mediators export an interface to their clients that is identical to that of translators. Both translators and mediators take as input SQL or OQL like queries and return OEM objects. We must note that OEM is a simple self-describing (tagged) object model for translators and mediators, allowing to refine in some way information from one or more sources. Hence, end users and mediators can obtain their information either from translators and/or other mediators. This approach allows new sources to become useful as soon as a translator is supplied, it allows mediators to access new sources transparently, and it allows mediators to be "stacked", performing more and more processing and refinement of the relevant information.

The user writes a query on an interactive HTML page, or selects a query from a menu using a Web browser. The answer is received as a hypertext document translating OEM objects in HTML. The root of this document shows one or more levels of the answer object, with hypertext links available to take the user to portions of the answer that did not appear on the root document.

Since the implementation of a mediator can be a quite complex and time-consuming task, one of the TSIMMIS goals is to provide the necessary tools for automatic or semi-automatic generation of mediators (see the *mediator generator* box on the right side of figure 10). Similarly, TSIMMIS provides appropriate translator generators that can generate OEM translators based

[26] http://www.isi.edu/sims/
[27] http://rodin.inria.fr/disco/
[28] http://www.cs.washington.edu/research/projects/softbots/www/softbots.html
[29] http://www-db.stanford.edu/tsimmis/

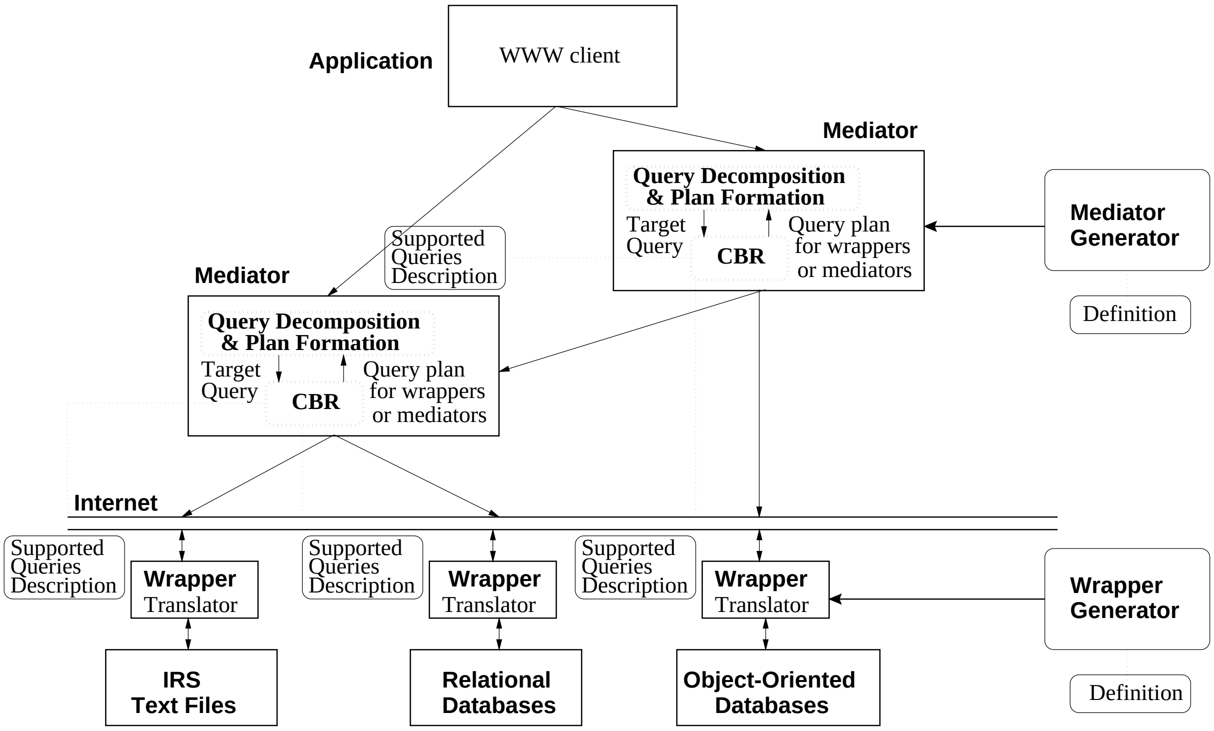

Figure 10: Architecture of the TSIMMIS

on a description of the conversions that need to take place for queries received and results returned (see the *translator generator box* on the right side of figure 10).

Finally, since the various information sources provide not only heterogeneous data but also different levels of query capabilities (IRS based, or DBMS based), TSIMMIS mediators ensure that sources receive only queries they can handle, while taking advantage of all of the query power of the sources. A key component of the mediator architecture is the *Capability Based Rewriter* (CBR). The CBR takes as an input a description of the capabilities of the data source, and a query targeted for that data source [VP97]. From these, the CBR determines component queries to be sent to the sources, commensurate with their abilities, and computes an execution plan for combining their results (using joins, selections, projections and unions). The CBR uses a description of each wrapper's ability, expressed in a special purpose language (Datalog like), to develop a plan for the wrapper's target query.

5.2.2 The Information Manifold project

The Information Manifold system[30] (AT&T Labs Rresearch) [KLSS95, LRO96] provides uniform access to multiple structured information sources on the World Wide Web (e.g., databases, form-based sources, etc.). As such, the system can answer complex user queries that require the combination of information from a large number of sources which can be frenquently changing. As in the case of TSIMMIS there is no global database schema, but the user poses queries in

[30] http://www.research.att.com/ levy/imhome.html

terms of a *world view* which is a collection of virtual relations and classes in a common model describing the contents of the sources (non materialized views).

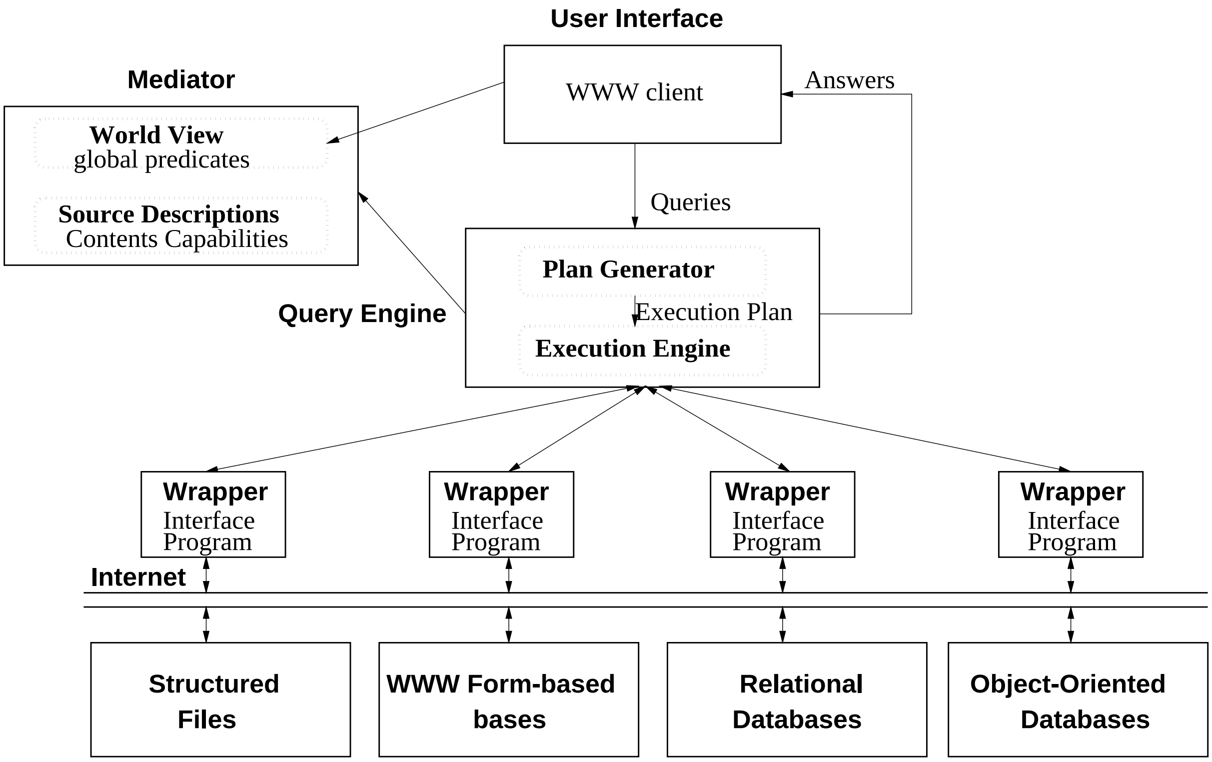

Figure 11: Architecture of the Information Manifold

The key aspect of the system is that it provides a source independent, query independent mediator component. Instead of being tailored to specific information sources and/or specific queries the input to Information Manifold is a set of descriptions of the contents and query capabilities of the sources using the *world view*. Given a query, the system uses the descriptions to determine which sources are relevant, sends the appropriate sub-queries to the relevant sources, and combines information from multiple sources to answer the user query. Consequenlty, we do not have to modify mediators each time a new information source is added or an old one is deleted. It should be noted that the goal of Information Manifold is to provide only a uniform query interface, and not update or transaction facilities.

As we can see from figure 11, the actual interaction with the information sources is done through *interface programs* (or *wrappers*). Logically, for every information source there is an interface program that accepts any query template available at the source and returns the appropriate answer.

The *mediator* component, apart from the commonly defined *world view* of the underlying information, contains declarative descriptions of the contents and the query capabilities of the data sources. More precisely, the contents of a source is described as a query (view) using the implicated *world view* relations and the conditions the tuples in the relation must satisfy. The query capabilities descriptions of a source allows to specify which parameters can be given in a query: the input attributes, the minimum and maximum number of inputs allowed, the possible outputs of the source and the selections the source can apply.

The *query engine* considers now the source descriptions to determine the relevant information sources as well as the query capabilities of the sources to prune the sources that are effectively accessed to answer the query.

To conclude, Information Manifold makes it quite convenient to add new sources. One must write a wrapper for the sources and define its views and constraints in terms of the global concepts. However, no change to the query-processing algorithm is needed. The new views will be used whenever they are appropriate for the query. In contrast, new TSIMMIS sources not only must wrapped, but the mediators that use them have to be redefined and recompiled. Of course, if the semantics of the local source has not been captured yet in the “World view”, an extension of the later is required.

5.3 Protocols for Distributed Systems

5.3.1 Z39.50/SQL+

Z39.50/SQL+ is an extension of the existing Z39.50-1995 (Version 3) protocol, uniting the advantages of SQL's query language and export syntax with the information retrieval services of Z39.50. The SQL extension provides a standardized way of specifying complex structured queries which otherwise are not possible with the existing Z39.50 query types. In addition, result sets (which may contain complex data types) may be returned in a generic record syntax without having to tag each individual field of each record and without being tied to a pre-defined schema. The Z39.50/SQL+ standard remains platform and database independent. Indeed, the supporting database may be a Relational DBMS, an object DBMS, an object-relational DBMS or even a text database for that matter. As it is now the case with Z39.50, the onus is on the server to map the Z39.50 PDU's to the applicable database calls.

Z39.50/SQL+ clients are able to specify complex queries, either by using SQL or one of its derivatives, such as Query-by-Example (QBE). Queries can be formulated on single (virtual) tables, or multiple tables supporting cartesian product, union, intersection, join on matching columns, and projection on given columns. Queries can also be formulated using powerful constructs for expressing conditions, performing aggregate and comparison operations, partitioning tables into groups and much more.

Z39.50/SQL+, like Open Database Connectivity (ODBC) and Remote Database Access (RDA) Protocol Part 2 - SQL Specialization, uses SQL as its standard way of expressing complex queries on structured data. However, it should be noted that the functionality of Z39.50/SQL+ is conceptually different from that of ODBC and RDA. The two latter simply convey SQL statements (both retrievals and updates) and related control operations (including concurrency control and synchronization) to the database management system. In contrast, Z39.50/SQL+ is concerned with interoperable information retrieval, and facilitates state management within and across communication sessions, a standard means of dynamically accessing the existing catalogues, the option of using a standardized schema for interoperability, as well as extra facilities such as scan and extended services.

In summary, Z39.50/SQL+ provides additional querying and retrieval power to existing Z39.50

implementors. It expands Z39.50 capabilities to a community much broader than the library-type community employing a wider class of databases as part of their information management resource bases.

5.3.2 Distributed object and document management

Accessing documents on the Web is done mainly by following links in order to access documents that are transferred using the HTTP protocol, either by directly fetching the document referred with an Url, or through the execution of a CGI script which retrieves or calculates the information from a server or a database. In spite of the simplicity of HTTP and CGI programs in designing interfaces for database remote access, this approach is too limited for advanced transactions due mostly to the stateless character of the protocol.

Due to Internet generality, freedom and heterogeneity, the developers should define and implement new protocols and tools for carrying specific tasks but keeping in mind the prerequisite of a quick integration into complex hypermedia applications for immediate, spread and possibly brief usage[REM+95]. More radical approaches foresee the complete redefinition of the information organization adopting object oriented technologies to connect the components of a generalized document that are distributed around the net.

Distributed Object Computing (DOC) based approaches lead to realizations which are more advantageous than conventional server-centered information systems in terms of flexibility, scalability, portability and robustness [SV95] but are currently expensive and difficult to develop. Among them, the Common Object Request Broker Architecture (CORBA) [Inc93], proposed by the Object Management Group (OMG), is having a certain success. In [REM+95] an attempt has been described for integrating CORBA with the Web.

Object Request Brokers (ORBs) specification provides the mechanisms by which objects transparently make requests and receive responses. The similarities with HTTP offer an opportunity to interoperate between CORBA based systems and the World Wide Web, and the challenge is to combine their strengths, rather than to compound their weaknesses. Distributed objects bring advantages in software development due to ease of programming, extensibility and managability, data and programs encapsulation and integration. Furthermore, CORBAs philosophy pays attention that the new tools fit with what already exists, that software is made freeware and that the already existing work is exploited. A more extreme position is assumed by HORB (HIRONA Object Request Broker) [Sat] which exploits Java's desirable features for DOC offering tools to develop client - server architectures with an object oriented approach, users have to deal with application objects which interact each other through socket interfaces.

Based on the features offered by this software, in [Dua96] a methodology is proposed for developing Java applications which manages database distributed on the Web. Request brokers supplied by HORB handle database systems interconnection based on client/server architectures developed in Java while offering flexibility, portability, scalability, robustness and object mobility in the development of DB applications.

6 Querying and Integrating Heterogeneous Data: Future work in the context of Aquarelle

We have insisted above on the need for a better exploitation by the user of the semantics of the data sources. We suggest in section 6.1 three specific solutions taking advantage of the current state of the art in uniform access to heterogeneous and distributed data sources. We then suggest in section 6.2 the use of geo-referenced documents and maps as an help to the user in his/her choice of the relevant documents. Finally section 6.3 opens the way to future work based on the use of a Knowledge Base System for user query formulation and mediation at the Access Server.

6.1 Integrating heterogeneous data

We have seen in section 5 two approaches for querying and integrating heterogeneous databases: a) multidatabase/federated approach and b) integration of heterogeneous and (semi-)structured sources using logical views. A third one is to query information whose structure is incompletely known and can be used in complement to the previous approaches. Some effort has to be devoted to look in more details which features of these general approaches have unsufficiently addressed by the Aquarelle community and is applicable to this context. We give below three specific solutions for a better user exploitation of the semantics available in the data sources.

The first one uses a language called POQL [ACC+97] developed at INRIA which is a first step toward querying data without complete knowledge of their structure. The fact that the structure does not have to be totally specified allows for integrating at a certain extent several sources which do not have the same structure. The power of this approach to query both structure and data at the same time using the same paradigm (data model and query language) is illustrated by the INRIA demo [31] on a database of the french Inventaire whose documents obey the SGML CI DTD. In the framework of Aquarelle we believe that a POQL-like approach will be more helpful for querying unknown structure than for actually querying heterogeneous SGML folder servers. This is developed in section 6.1.1.

The second one is based on the multidatabase approach. The demo[32] developed at CNR illustrates this approach. It is based on a stronger schema integration and is more suitable to managing and querying existing archive servers. The use of a strong, controlled schema integration implies that adding a new data server is less trivial, but has for a counterpart a better description of the semantics of the information sources and thus its use at a user level and at a conceptual level. This scenario suitability for adapting to heterogeneous data source querying such as in Aquarelle has to be tested. This is developed in subsection 6.1.2.

The third one will not be described here. It corresponds to the TSIMMIS approach (see section 5). It champions the use of a simple generic semi-structured data model, OEM. This implies the definition of the protocol between the data sources and the access server, the definition of data source wrappers according to OEM and the latter protocol, the redefinition of the access server according to this model and an integration of the metadata with the OEM schema. An

[31] http://cosmos.inria.fr:8080/poql.html
[32] http://africa.itim.mi.cnr.it

application to Aquarelle european cultural heritage implies the adoption of a common structure description language such as SGML or XML for the data sources. Compared to a hardwired solution such as the current one extended to take into account for example the metadata knowledge, it provides an elegant, adaptable and simple model for organizing, querying both, on the one side metadata and schema, and data on the other side.

6.1.1 POQL as an alternative to Access Points based queries

As we have seen above, extraction, discovery of the structure, browsing through the structure is fundamental for the user to formulate his/her query, eventually refine it and send simple IRS queries. We have suggested that to querying the schema (i.e., the structure of documents) of a specific data server might help the user in his query formulation process. The idea now is that the user does not know in advance which servers to query, and would like to query the structure of several unknown servers. In other words, *this schema querying should be independent of the data source.* We need a uniform way of querying the structure of heterogeneous sources.

One good candidate for such an approach which queries both structure and data at the same time is POQL [ACC+97, CCM96, CACS94] developed at INRIA on top of the object-oriented DBMS O2. We give here a simple application of POQL for SGML-encoded folders stored in ODBMSs (e.g., SIS), as an alternative to the current Access Points based queries. Instead of artificially mapping the folders structure semantics onto Aquarelle Z39.50 Access Points, one might use advanced query languages like POQL for APs based queries.

In this context, we can solve an AQL/Z39.50-like query by looking for all folder fragments containing a given value of an Access Point. Such a query, whose result could be a specific folder fragment or the whole folder, would follow unspecified or semi-specified paths in the document whose structure depends on the data server. POQL typically allows for answering queries the path of which in the structure is unknown. For instance, supposing that a user wants to find all folders containing Cognac in their title, the following query must be issued:

```
select  f
from    Folders{f}@P.#A(x)
where   x contains "Cognac" and name(#A) contains "tl"
```

where *Folders* is the name of a folder server database, *f* is a variable ranging over the folders, *@P* is a path variable allowing to express navigation through the unknown structure of folders, and #A a variable ranging over the attributes ending the paths. Then, the filtering condition specifies that the required attributes (i.e., the Access Points) contain "tl" in their name and the corresponding values x (i.e., Access Points values) contain the string "Cognac". This is logically equivalent to the definition of an Access Point *title* in the Aquarelle Z39.50 profile and its corresponding mapping to the related elements (i.e., SGML tags) of folders as for instance `tl-cl` (for "classeurs"), `tl-th` (for "thematics"), `tl-dos` (for "dossiers"), `tl-obj` (for "objets") in the CI DTD.

The advantage of POQL is that one does not have to specify in the query all possibilities (tl-cl, tl-th, tl-dos, etc..) that we do not have to specify the paths to access to those "classeurs", "thematics", etc. Furthermore, and this is more important, the POQL queries are in a certain degree independent of the data structure. This has at least three advantages:

1. querying multiple heterogeneous data sources is possible: a single POQL query can be run against several sources. But some translations of the names used in POQL queries (e.g. *Folders* in the above query) are required for each data source.

2. integration of new data sources to the system is easy: no mapping is foreseen between the APs and the source structure. But of course the data source has to interpret POQL queries.

3. if any data source changes its structure (e.g. if the CI DTD changes), the same queries are still valid.

Once the list with the relevant folders is returned to the users, they can start navigating deeper in the folder structure and content (with an HTML/SGML browser) to locate specific parts of interest. We claim here that query languages like POQL allow to more easily retrieve not only fine grain document fragments but also the *structured context* occurring in a folder. For instance, in order to find the relevant *title* fragments as well as the corresponding structure, i.e. the paths from the root of the folders to the title, users may refine the above query as follows:

```
select  f, @P, #A, x
from    Folders{f}@P.#A(x)
where   x contains "Cognac" and name(#A) contains "tl"
```

This is extremely useful to cope with the problem of "orientation" within a vast space of interconnected folders or folders fragments.

In addition, as suggested above (see section 3.3), if the folder servers were to store together with the SGML documents the associated hyperlinks (actually residing only at the Access Server) more sophisticated queries on these interconnected graphs of SGML documents can be expressed. For instance, we can find the folders having somewhere a reference to another folder concerning "Cariatides". Such queries are clearly out of the scope of the current Aquarelle architecture.

In summary, the above use of POQL illustrates a possible better integration between the Access server and the data servers which does not require a "hardwired" static mapping between Access points and the data source schema. On the contrary it allows for evolutivity of the data server without affecting the configuration of the Z39.50 gateways or the Access server. Future work in this direction should investigate the integration of POQL queries with the Z39.50 protocol (in the spirit of Z39.50/SQL+) and the implementation of a subset of POQL retrieval features also for relational DBMS (possibly using appropriate Explain Databases). Finally, as will see in the next section, an interesting issue is to use POQL at a global schema (multidatabase) level integrating multiple heterogeneous local data structures and generate a number of structured queries tailored to the local data sources.

6.1.2 Multidatabase Perspective for Aquarelle

As described in the state of the art section, the Common Database Model (CDM) has a key role in multidatabase systems: it has to be both rich enough to capture local schemas' semantics and simple enough to facilitate either the creation of the federated schema or the query formulation in the multidatabase language.

The question is whether the multidatabase architecture and the adoption of a CDM-like component are adapted to situations where local information repositories are not always managed by DBMSs and their structural aspects are weak or not completely explicit. As the CDM is usually implemented and maintained either by a global administrator or by the cooperation of the local administrators, when applied in the Web environment the solution presents a serious drawback: it is very hard to maintain the relationships between the global and the local schemas.

However, when used to connect heterogeneous information repositories in a proprietary client/server environment as in the Aquarelle project, the idea of CDM survives in its essential features that is, archive content cataloging, control of user navigation, and query formulation guiding. To reach this goals the communication of metadata on archive content is not sufficient: on the one side the server needs non elementary tools to describe both semantics and structure and let them emerge to the client side; on the other side, the client must be provided with functional components able to build and maintain a complex global description of the knowledge emerged from archives. The effort is in building a global representation which is not a simple aggregation or a mapping of the available access points in the different local sites : the global component should follow a model powerful enough to represent more complex cases resolving overlapping semantics between sites and retaining the minimal meta information extracted by less structured archives.

Solutions derived from object-oriented CDMs may be adopted (see the GARLIC system in subsection 5.1). In fact, some of their characteristics, which are discussed in [PBE95] look particularly attractive also when applied in the strongly heterogeneous and autonomous Aquarelle environment. Furthermore, the global component has to provide to the User Interface functionalities to guide users in selecting most interesting repositories and to exploit dynamic presentations both of access mechanisms and of facilities to better express and refine queries.

We believe in the usefulness of a common on-line classification to be adopted by the user as well as by the database administrators when they describe the archives they manage. As an example, the Information Coding Classification (ICC), which is available on-line[33], could be exploited as a browsing support allowing the user to select a set of archive subjects of interest.

Therefore, the emergence of archive descriptions at a global level, archives selection through an on-line classification, transparent querying through archive common domains, form the framework of a promising tool. Such a tool developed at CNR and demonstrated in http://africa.itim.mi.cnr.it is currently being extended. It allows semantic overlapping of local archives with different structures and management of systems which interoperate through Internet.

[33] http://www.isrds.rm.cnr.it/ITA/cirt/cirt.html

6.2 Geo-Referenced Navigation and Querying

In many culture heritage applications, folders are related to geographical areas, i.e. they are geo-referenced. Depending on the scale the geo-reference might be a point (Lambert coordinates) or a zone (cadaster scale). The use of geo-references and the association of folders to geographical maps is useful for at least two reasons:

- user interface, navigation, query refinement : instead of access point based access to information, the user might want to navigate through a geographical area zooming from a country scale down to a county scale before deciding which folder(s) to access. At each scale, points featuring folders are displayed on a background map
- querying : as in Geographical information systems (GIS), the user might want to join to the usual search criteria, spatial ones : " give me the folders associated with the 18th century farms located within 5km from Cognac"

The current experiment by the French Ministry of Culture (Inventaire) jointly with INRIA and Euroclid aims at prototyping the access through the web to geo-referenced folders structured according to the CI SGML DTD including the two above features. With the help of administrative maps from the french national Institute of Geography (IGN), the user will be able to narrow down his choice in browsing through maps and choosing a geographical area of interest on which are displayed points representing folders. The scale will go down till the cadaster level and the user will be able to display the cadaster vectorial geometry representing say a building, the parcel around, etc. Whatever the scale of the background map, the user will be able to issue queries combining traditional text or access point like predicates with spatial predicates. The results of this experiment should be looked at in the Aquarelle context.

6.3 Towards KBS-based Information Integration

The diversity of users, the diversity of quality required in the answer to a query, have not been taken into account in the current Aquarelle design. It has also been argued that a given domain knowledge should be more systematically exploited both at the user level and at the mediation level. We have also claimed that the data source semantics is unsufficiently taken into account. Most of the current effort to take into account this semantics is through the exploitation of the sources metadata.

This is only a first step towards the definition, acquisition and exploitation of knowledge related both to domains of interest to the users, and to the data sources integrated. We plan to investigate an architecture in which a Knowledge Base System (KBS) would be coupled to the Access Server as well as to the GUI with the following functionnalities : (i) helping te user in his query formulation depending on his/her expertise level as well as on his/her domain of interest; (ii) helping the Access Server in his mediation with the data sources (choice of the relevant data server, reformulation of the query, etc.).

Another important feature of such a KBS based architecture is knowledge acquisition and maintenance, according to new data source integration or existing source update, knowledge discovery in unstructured data source as well as knowledge base maintenance related to new domain integration or refinement of existing domains.

Acknowledgments: We are grateful to Martin Dörr for fruitful discussions and Irene Fundulaki for helping in the editorial work.

References

[ACC+97] S. Abiteboul, S. Cluet, V. Christophides, G. Moerkotte, and J.Simeon. Querying Documents in Object Databases. *Journal of Digital Libraries*, 1(1):5–19, April 1997.

[BHP92] M.W. Bright, A.R. Hurson, and S.H. Pakzad. A taxonomy and current issues in multidatabase systems. *IEEE Computer*, pages 50–60, March 1992.

[CACS94] V. Christophides, S. Abiteboul, S. Cluet, and M. Scholl. From Structured Documents to Novel Query Facilities. In *SIGMOD'94*, pages 313–324, Minneapolis, Minnesota, USA, May 1994.

[CCG+97] P. Carrara, A. Crippa, C. Ghiselli, A. Licciardi, D. Musella, M. Padula, and L. Tonelli. Heterogeneous document management in a virtual library. Technical Report ITIM006/97, ITIM, 1997.

[CCM96] V. Christophides, S. Cluet, and G. Moerkotte. Evaluating Queries with Generalized Path Expressions. In *SIGMOD'96*, pages 413–422, Montreal, Quebec, Canada, June 1996.

[CDF97] V. Christophides, M. Dörr, and I. Fundulaki. A Semantic Network Approach to Semi-Structured Documents Repositories. In *ERDL'97*, Lecture Notes in Computer Science, Pisa, Italy, September 1997. Springer-Verlag.

[CF96] R. Colomb and S. Finnigan. Z39.50/SQL+ - Profile for SQLAccess in Z39.50, September 1996.

[CGMH+94] S. Chawathe, H. Garcia-Molina, J. Hammer, K. Irelandand Y. Papakonstantinou, J. Ullman, and J. Widom. The TSIMMIS Project: Integration of heterogeneous information sources. In *IPSJ*, Tokyo, Japan, October 1994.

[DBF96] M. Dörr, C. Bekiari, and J. Farsaris. Folder server and editor: Technical specifications. Technical report, ICS-FORTH, September 1996. Deliverable D6.1 of the Aquarelle Project.

[DF97] M. Dörr and I. Fundulaki. A conceptual model for the representation of multiple interlinked thesauri. Working Paper #26, ICS-FORTH, March 1997.

[DSWM96] D.A. Duce, D.C. Sutcliffe, T.A. Watson, and D.F. MacRandal. Aquarelle Technical Specifications-Version 1. Technical report, RAL, July 1996. Deliverable 3.1 of the Aquarelle Project.

[Dua96] N.N. Duan. Distributed Database Access in a Corporate Environment Using Java. In *Fifth International WWW Conference*, Paris France, May 6-10 1996. http://www5conf.inria.fr/fich_html/papers/P23/Overview.html.

[Fah94] G. Fahl. *Object views of relational data in Multidatabase Systems*. PhD thesis, Dept. of Computer and Information Science, University of Linkping Sweden, 1994. Thesis No. 446, http://www.ida.liu.se/ gusfa/Papers.

[Fur96] N. Furh. Object Oriented and Database Concepts for the Design of Networked Information Retrieval Systems. University of Dortmund, 1996.

[HM85] D. Heimbigner and D. McLeod. A federated architecture for information management. *ACM Trans. on Office Information Systems*, 3(3), July 1985.

[Inc93] Object Management Group Inc. The Common Object Request Broker: Architecture and Specification, December 1993. Revision 1.2 OMG Document Number 93.12.43.

[KLSS95] T. Kirk, A. Levy, Y. Segiv, and D. Srivastava. The Information Manifold. In *AAAI Spring Symposium Series on Information Gathering from Heterogeneous, Distributed Environments*, Stanford University, March 1995. http://www.isi.edu/sims/knoblock/sss95/proceedings.html.

[LMR90] W. Litwin, L. Mark, and N. Rousopoulos. Interoperability of multiple autonomous databases. *ACM Computing Surveys*, 22(3):267–293, September 1990.

[LRO96] A. Levy, A. Rajaraman, and J. J. Ordille. Querying heterogeneous information sources using source descriptions. In *VLDB'96*, pages 251–262, Bombay, India, September 1996.

[Mar97] A. Marrara. The Aquarelle User Client - Version 1. Technical report, Finsiel, June 1997. Deliverable 4.7 of the Aquarelle Project.

[PBE95] E. Pitoura, O. Bukhes, and A. Elmagarmid. Object Orientation in Multidatabase Systems. *ACM Computing Surveys*, 27(2):141–195, June 1995.

[REM+95] O. Rees, N. Edwards, M. Madsen, M. Beasle, and A. McClenaghan. A Web of Distributed Objects. In O'Reilly & Associates, editor, *Proceedings of the Fourth International WWW Conference*, pages 75–87, December 1995. http://www.w3.org/pub/Conferences/WWW4/Papers/85/.

[RS97] M. T. Roth and P. Schwarz. Don't scrap it, wrap it! A Wrapper Architecture for Legacy Data Sources. In *Proc. of the 23rd VLDB Conference*, pages 266–275, Athens, Greece, August 1997.

[Sat] Hirona Satoshi. Horb, etl. Japan. http://ring.etl.go.jp/openlab/horb/.

[SL90] A.P. Sheth and J.A. Larson. Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. *ACM Computing Surveys*, 22(3):183–236, September 1990.

[SLS97] O. Signore, M. Loffredo, and S. Sabina. Z39.50-SQL gateways: techical description. Technical report, CNR-CNUCE, March 1997. Deliverable 5.1 of the Aquarelle Project.

[SV95] D.C. Schmidt and S. Vinoski. Object Interconnections, Introduction to Distributed Object Computing. *SIGS C++ Report*, 7(1), January 1995.

[Ull97] J. Ullman. Information Integration using Logical Views. In *ICDT'97*, Delphes, Greece, January 1997.

[Vas94] D. Vaskevitch. Database in Crisis and Transition: A Technical Agenda for the Year 2001. In *SIGMOD'94*, pages 484–489, Minneapolis, Minnesota, USA, May 1994.

[Ver97] A.M. Vercoustre. Aquarelle, WG-6 some issues about metadata. Technical report, INRIA, January 1997. http://www-rocq.inria.fr/ vercoust/PAPERS/metadata.html.

[VP97] V. Vassalos and Y. Papakonstantinou. Describing and Using Query Capabilities of Heterogeneous Sources. In *Proc. of the 23rd VLDB Conference*, pages 256–255, Athens, Greece, August 1997.

[Z3996] Z39.50 Profile for Access to Digital Collection. Library of Congress, May 1996.