An architecture for accessing a large number of autonomous, heterogeneous databases

Sergio Flesca, Luigi Palopoli, D. Saccà, D. Ursino

DEIS
Università della Calabria
87036 Rende (CS), Italy
{flesca,palopoli,sacca,ursino}@si.deis.unical.it

ABSTRACT. In this paper an architecture supporting Internet-based integrated access to heterogeneous, autonomous distributed databases is illustrated. The proposed architecture is two-tier and can be seen as a variant of the well-known mediator-based one. The main feature of this architecture is that it is specifically conceived to realize very large systems, comprising hundreds of local databases. A structured, hierarchical global scheme, called data repository, is adopted as the core structure to guarantee friendly access to available data through a graphical Web interface. The architecture described here has been implemented in a system, called NET R, constructed on top of the set of databases owned by Italian central Government offices.

RÉSUMÉ. Cet article propose une architecture offrant des accès intégrés à des bases de données distribuées, autonomes et hétérogènes via l’Internet. L’architecture proposée est two-tier. Elle peut être vue comme une variante de l’architecture basée sur les médiateurs. La principale caractéristique de cette architecture est qu’elle est conçue spécialement pour des systèmes importants, pouvant comprendre des centaines de bases de données locales. Un schéma global hiérarchique et structuré, appelé le référentiel des données, est la structure centrale qui garantit des accès aisés aux données à travers une interface graphique Web. L’architecture a été implémentée dans un système (appelé NET_R) construit au-dessus d’un ensemble de bases de données du gouvernement central italien.

KEY WORDS: Cooperative Information Systems, Internet-based information systems, heterogeneous data sources, database integration, data repository.

MOTS-CLÉS: Systèmes d’information coopératifs, Systèmes d’information basés sur l’Internet, Sources de données hétérogènes, Intégration de bases de données, Référentiel des données.

Introduction
The issue of integrating geographically spread, heterogeneous databases has been dealt with since early seventies and has now received a renewed interest because of the recent technological development of networks and of the availability of numerous data sources.

Multidatabase systems can be considered as the first attempt to realize the integration of existing databases [BRE 90]. They achieved connection between DBMS by adopting a strong integration approach. Due to this choice, multidatabase systems had necessarily to rely upon a complex and sophisticated software support for communication and information exchange. Besides the implementation difficulties caused by this complexity, these systems were also characterized by important organizational problems in their management, mainly determined by the loss of autonomy for pre-existing databases.

Federated databases [LIT 90, SU 96, WIE 92, SHE 90, THO 90] have been proposed more recently to overcome these limits. Federation requires a much looser integration amongst sites so that the autonomy of local databases is mostly preserved. Indeed, in this context, the integration step consists in designing interfaces as software layers upon existing DBMS to coordinate and integrate underlying databases without modifying either their structure or their internal behaviour. Therefore, in this case, the integration software is simpler for it basically provides support for information exchange and information sharing amongst sites; also the organization is simpler since the autonomy of local systems is preserved. From a functional standpoint, such a structure can be looked at as consisting of information providers supplying their information and information users exploiting them, where each local database site can either act as a provider, or as a user or both.

Even if traditional federated DBMS constitutes an important step forward w.r.t. multidatabase systems, they still have an important drawback in that information flow is not really coordinated, thus possibly causing “dispersion” of information. Such a negative characteristic is negligible for relatively small systems, but becomes relevant when the amount of managed information is large. Unfortunately, especially when geographically dispersed information sources are to be integrated, the latter is the most probable case.

Therefore, the most relevant problem of federated systems towards the satisfactory exploitation of pre-existing information sources seems to be the absence of a sw/hw entity providing the needed information flow coordination amongst provider and user sites. Such an entity would, for instance, give users the opportunity to single out only those data necessary to their information needs, decide access rights to information and, more generally, choose all the presentation details relative to information supplied by provider sites. This entity can be thus
An architecture for accessing a large number of autonomous, heterogeneous databases

looked at as an information mediator [WIE 92, ULL 97] between information sources and information users.

In the last years, some mediator-based systems have been designed and realized. For example, the Information Manifold [LEV 96], developed at the AT&T Laboratories and Tsimis [GAR 95], designed at Stanford University, are examples of realization of mediator capabilities in integrated information systems. Mediator-based architectures are two-level structures, consisting of a wrapping layer and mediation one. Wrappers translate local languages, models and concepts of the data sources into the global language, model and concepts shared by some or all sources [WIE 92, ULL 97]. Mediators take in input information from one or more components below them, and supply as the output either information for the components above them or for external users of the system. Data are not necessarily localized at the mediator. Often the mediator stores just views of real data. These views can be queried by components above them (including system users): it is then up to the mediator retrieving data needed for constructing query answers from the actual data sources. A typical mediator based architecture is shown in Figure 1.

![Figure 1. Architecture of a mediator based system](image)

Tsimmis [GAR 95] is based on a Query Centric model, i.e. mediated scheme relations are defined as views over the source relations. The mediator describes
information contained in the various sources using a global scheme (so there is only one abstraction level); the global scheme is queried by the user in order to access information. In this system data models of the different local databases are translated into a unique one; the global data model is powerful enough to manage various information representations.

Information Manifold [LEV 96] is based on a Source Centric model, i.e., for the purpose of integration, the source relations are defined as views over mediated scheme relations. Therefore the global scheme is a collection of virtual classes and relations against which the user poses queries.

In both systems the global scheme specifies the mapping between virtual and real relations; in both of them querying takes place at the level of the global scheme and the mediator has the responsibility of constructing subqueries which are dispatched to local wrappers and whose results are collected and merged to form the answer to the user query.

Despite its certain merits, the standard mediator-based architecture illustrated above could be difficult to realize when the number of involved databases is very large (in the order of hundreds). Indeed, we argue, there are two main problems to be dealt with:

- The adoption of a “flat” global scheme seems not to be adequate to guarantee a reasonably structured and abstract description of available data. Indeed, it can be anticipated that in situations where hundreds of database schemes are to be integrated, the number and the complexity of relationships holding among involved database objects would be overwhelming. This situation would immediately cause a substantial difficulty for a user to retrieve the information he/she needs.
- The management of querying processes, potentially accessing hundreds of databases for each single user request, may become quite hard a task to be autonomously carried out by the system.

The architecture proposed here, which was explicitly designed with the purpose of supporting such very large integrated access systems, is intended to solve these problems. In order to achieve appropriate support to accessing many data sources (which is the fundamental purpose of our system), it adopts a structured and complex global scheme in the place of an ordinary “flat” global scheme. On the other hand, it lights the burden lying on the mediator modules, by letting the user to perform the selection of local data sources of interest in a friendly, guided manner. Thus, from the user perspective, querying consists of two distinct phases. In the former one, the user is guided by the system in “navigating” the structured global scheme in order to select a (set of) database(s). In the latter one, the user submits his/her query on the selected data source using a QBE-like query interface and gets corresponding answers.
In other words, in our architecture, the global scheme of mediator-based systems is substituted by a complex, hierarchical dictionary, called data repository, which stores several (partial or total) global views over available data, at different abstraction levels.

The data repository is primarily intended to provide a global view of data owned by an organization using an integrated, comprehensible, accessible and usable structure. Generally speaking, data repositories allow to understand how information is organized in local databases and indicate how databases describe the same information or similar ones, pointing out differences or similarities and correlations among representations, in that being an effective reference for designing new applications which must use data already present in the available database systems.

The design of a data repository is a complex process, consisting in iterating several phases and goes behind the scope of this paper.

The interested reader is referred to [BAT 95, BAT 84] for more about this subject. Figure 2 reports the data repository of Italian Central Governmental Office (ICGO) databases which has been recently designed by Italian Authority for Informatics in Public Administration (AIPA).

The ICGO data repository involves about 300 databases owned by various central administrations. Some of these databases have a relevant size. They are generally based on a variety of data models and systems. Databases are grouped into 30 clusters; each cluster encloses semantically homogeneous databases. Upper levels of the tree have been obtained by carrying out integration-abstraction phases on schemes of lower levels. Each node within the tree structure represents (a portion of) the integrated system of the ICGO at a certain abstraction level.

It should be clear enough that a data repository like the one illustrated above stores the necessary information for helping the user in locating and retrieving the information of interest, even if he/she has only a partial and abstract knowledge of the data available in the system.

The structure of our system, as embedded in the general framework of mediator-based integration systems, can therefore looked at as shown in Figure 3.

Note that, in our system, the wrapping modules exploit local database catalogues and are responsible of the translation from the data model and language of the mediator to the data models and languages of databases managed by them.

This architecture has been implemented in a prototype developed at DEIS and called NET_R. In accordance with our general architecture, NET_R is a two-tier distributed system that includes provider sites and one distributor site.
Figure 2. Structure of ICGO Data Repository

As explained in more details in the sequel, the distributor site stores the data repository and acts as the data mediator of the system, whereas provider sites act as wrappers. NET_R is an Internet-based system allowing for the integrated access to geographically dispersed databases. A Web interface exploits the information
encoded in the data repository description for helping the user of the integrated system in locating a local database of interest and in formulating queries on it. Querying is based on a simple QBE-LIKE graphical format: the system also provides for the needed translation from the graphically formulated query into the local system query language.

Main functionalities provided by NET R are listed below:

- dynamic construction of HTML and JavaScript pages for the visualization of repository schemes (including local database schemes of information provider sites);
- repository navigation for accessing underlying databases according to user's information needs;
- graphical query formulation support through a QBE-like interface in WWW format;
- management of user query submission and query result reception to/from remote provider sites (where the databases of interest are physically located);
- dynamic construction of HTML and JavaScript pages for the visualization of user query results.

Figure 3. Architecture of NET_R
Our system matches the application requirements in that:

- it adopts a structured global scheme with several abstraction levels and an interactive navigation through the mediator scheme; as a consequence the user can focus only on the data of interest and, consequently, he/she can retrieve information he/she needs without being puzzled by the presence of large amounts of uninteresting data;
- the querying process becomes very simple since the user queries the databases containing the information of interest and the system limits itself to considering only the data necessary for the answer.

We note that other systems having a similar purpose as NET_R, but neither based on a mediator scheme, nor explicitly designed to integrate a large number of local databases, have been proposed [AND 94, ULL 97, LIT 90, WIE 92]. An interesting example is the system Mosaic [AND 94]. This system (and its descendants) hide from the user the specific protocols associated with contacting each information source. However the user needs to know about an information source in order to access it. The architectures presented in [ULL 97, LIT 90, WIE 92] are based on semantically rich data models for representing semantic connections between distributed data but they focus on integrating relatively small number of databases.

Before closing this section, it is worth pointing out that although NET_R is based on the ICGO data repository, it could alternatively exploit the structure information stored in any other data repository.

The rest of the paper is organized as follows. In the next section we will describe the general architecture of NET_R. In Section 2, system functionalities are illustrated. A more detailed account of the structure of mediator and wrapping modules is given in Section 3. In Section 4, the Web user interface is briefly described. Finally, in Section 5, we draw our conclusions.

1. General architecture

The NET_R system includes two kinds of site, called the provider site and the distributor site, respectively. The distributor site stores the data repository and acts as a data mediator in our system. It receives user information requests and process them, supplying results in the appropriate form back to the user. Note that, in our system, users are only allowed to perform data querying: both data updating and scheme updating are forbidden to them.

The behavior of the system, as illustrated in the context diagram of Figure 4, is the following. The user wishing to formulate a query activates a connection to the distributor site through a WWW browser. The distributor site manages both
An architecture for accessing a large number of autonomous, heterogeneous databases

submission of the query to the provider site where the database that was selected by the user is located, and corresponding answer collection and displaying.

If the time slot necessary to obtain an answer from the provider site exceeds a given time-out, an off-line answer management process is activated at the distributor site and the user is notified disconnection due to time-out. The activated process asynchronously collects all the query results relative to timed-out connections.

The user is also allowed to formulate off-line queries, and to require the distributor site to submit them. In general the distributor site submits queries to the appropriate provider site manager, which translates the query into the local DBMS language and executes it. The constructed answer is collected by the provider site manager, and sent to the distributor site manager which, in turn, displays it to the user. Usually, the user obtains the answer to a submitted query on-line. However if the asynchronous communication mechanism is to be used, the distributor site manager stores query results so as to be available to the user to have it displayed during following sessions.

The system also allows to program predefined “parameterized” queries, corresponding to most common requests made by users and which users “complete” by specifying input parameters.

As far as the NET_R user interface is concerned, we have already mentioned that it runs within any WWW browser. The connection is established towards the distributor site and is, as already noted, generally synchronous in that the user remains connected to the system through the WWW browser until to he/she decides to disconnect the session, or a disconnection due to time-out is forced.

Let us now give a few remarks on updates. We have already pointed out that users are not allowed to carry out updating neither on data nor on schemes. However, such updating can take place at the level of local databases. In particular, we have to consider two different cases as far as the modification of local database schemes is concerned. If the modifications carried out locally do not induce changes in the repository structure, then the involved provider site manager automatically sends all data necessary to the modification of the global dictionary to the distributor site manager. Vice versa, when the system is faced with a major modification, such as the insertion of a new database, which can change the repository structure, the intervention of the administrator of the global dictionary is required. Indeed, first of all, a notification message is automatically sent to the distributor site; then the new database is inserted and the administrator of the global dictionary is required to brought about appropriate modifications to the data repository. The administrator of the global dictionary accesses the repository through the distributor site and requires the information needed for activating the procedures concerning its modification.
A similar process is executed by the administrator of exported scheme dictionary to manage local database schemes which are globally visible in the system.

**Figure 4. Context Diagram of NET_R**
An architecture for accessing a large number of autonomous, heterogeneous databases

Figure 5. First Level Abstract Scheme of the ICGO repository

1. System Functionalities

As pointed out above, NET_R uses the data repository as a tool for navigating the scheme hierarchy represented therein to facilitate user selection and access to underlying databases. Once the user has singled out his/her database of interest, the system helps him/her to query this database, through a WWW QBE-like graphical query interface.

In more details, our system allows a user, say U, to navigate through the data repository according to four different methods:

- **direct access**: in this case clusters grouping local databases which U is allowed to access are visualized. A graphical interface allows U to choose the cluster of interest. As a consequence, the system presents a list of databases associated to the selected clusters that can be queried.

- **hierarchical access**: in this case U navigates the data repository through schemes. In any moment, the system maintains a "current scheme" and displays the list of its subschemes among which the user selects the "new" current scheme. Initially, the current scheme is the First Level scheme (see Figure 2).

- **access based on sources**: in this case U singles out a source of interest amongst data sources recognized in the system. Once this choice has been made, the system displays the list of databases associated the selected source.

- **object-guided access**: this is the most important navigation method in that it exploits the repository as a guide for searching the database of interest. This type of access is typically exploited by users who have a just general
knowledge about the information of interest but do not know the details of the information structure and localization. At first the system displays objects belonging to the most abstract scheme within the repository. Among the displayed objects, the user then selects that object $O$ which is most closely related to his/her information need. As a consequence, the system will display objects, belonging to the next level schemes of the repository, from which $O$ has been derived through integration and abstraction steps. The process is iterated by selecting a further object and displaying the associated set of objects in the next level.

Figure 6. Second Level Abstract Scheme of the ICGO repository

As an example, consider the ICGO repository in Figure 3; its first, second and third level abstract schemes are illustrated in Figures 5, 6 and 7. At the beginning the objects Good, Document, Subject, Office, Territorial References, Event and Public Services are displayed. Suppose the user chooses the entity Subject. The system will therefore display the set of objects $S$ of the second level abstract scheme such that Subject has been obtained by abstracting/integrating over $S$; these are Physical Subject and Non Physical Subject. Suppose the user chooses Non Physical Subject: the system displays the objects of the next lower level from which Non Physical Subject has been derived through integration and abstraction steps; these objects are Foreign State, Firm, Organization, Ministry. The process is further iterated and terminates when the level of database schemes is reached, where the user can submit the query of interest.
3. Structure of mediator and wrapper

3.1. Mediator

In our architecture mediation functionalities are carried out by the distributor site; its purpose is that of providing location transparency on databases managed by various provider sites. It carries out two fundamental tasks:

- it manages user access rights to information stored in local databases;
- it reconstructs information supplied by information providers in suitable aggregated structures within the data repository which are used, as previously described, to help the user in accessing the appropriate information sources.

![Third Level Abstract Scheme of the ICGO repository](image-url)
Table 1. Attributes of entities of Global Dictionary Scheme

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Name</th>
<th>Description</th>
<th>Abstract</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract Scheme</td>
<td>Level</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Database Scheme</td>
<td>Source</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Entity</td>
<td>ID</td>
<td>Name</td>
<td></td>
</tr>
<tr>
<td>Group</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Provider Site</td>
<td>ID</td>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>Database</td>
<td>ID</td>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>Relation</td>
<td>ID</td>
<td>Name</td>
<td>Priority</td>
</tr>
<tr>
<td>Attribute</td>
<td>Name (1)</td>
<td>Key</td>
<td></td>
</tr>
<tr>
<td>Query</td>
<td>Name (2)</td>
<td>Submission Date</td>
<td>Arrived Answer</td>
</tr>
<tr>
<td>Previously Define Query</td>
<td>Name (3)</td>
<td>Priority</td>
<td></td>
</tr>
<tr>
<td>Input Fields</td>
<td>Name (4)</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>User</td>
<td>login</td>
<td>priority</td>
<td></td>
</tr>
</tbody>
</table>

1. The key of “Attribute” consists in the attributes ID of “Relation” and name.
2. The key of “Previously Defined Query” consists in the attributes ID of “Database” and name.
3. The key of “Query” consists in the attributes login of “User” and name.
4. The key of “Input Fields” consists in the attributes ID of “Database”, name of “Previously Defined Query” and name.
The database which forms the core of the Distributor Site is a global dictionary in which all repository information are stored together with information about user access rights and actual data sources. The global dictionary is shown in Figure 8.

Attributes and entities shown in the scheme of Figure 8 are reported in Table 1 and their meaning is obvious. When the key of an entity consists just of one attribute, this is underlined; as for the other entities, the complete list of the attributes that constitutes the keys is given right after the table.

A brief description of catalogue content is given next. A scheme can be an abstract scheme or a database one. The entity \textit{database scheme} serves the purpose of describing database schemes, while the entity \textit{abstract scheme} describes repository schemes (see Figure 3). In the entity “abstract scheme” the attribute \textit{level} assumes a value equal to the distance of the abstract scheme from the root node of the repository. The only exception is constituted by clusters. To recognize clusters we have decided to use a conventional negative value. Remember that clusters are those structures to which databases are directly associated and have been obtained from their schemes through an integration-abstraction process. An abstract scheme contains one or more entities while an entity occurs just in one abstract scheme. A unique repository scheme is associated to each database and vice versa.

An \textbf{entity} can be a child of one entity and it can be a father of more than one entity (denoting a generalization process). On the other hand, an entity can be contained in a \textbf{group} and a group can contain one or more entities (denoting an abstraction process). The group, in turn, can be represented by only one entity, at the next upper abstraction level.

A \textbf{database} is associated to only one \textbf{provider site} whereas one of these sites can possess one or more databases.

The \textbf{user} can submit queries to databases. This entity has a \textit{priority} field: user can submit predefined queries and access relations with a priority less or equal than his/her own priority. User can access one or more abstract schemes and an abstract scheme can be accessed by one or more users.

The portion of scheme corresponding to the entities \textit{database}, \textit{relation} and \textit{attribute} has an obvious meaning and therefore is not further illustrated.

The architecture of the distributor site is shown in Figure 9. The \textit{Navigation Manager} realizes user navigation through the data repository. The \textit{Query Manager} is activated when the user requests to formulate a query. The module realizes a QBE interface. Information necessary to support query formulation is, of course, retrieved from the Global Dictionary. The \textit{TCP/IP Client} serves the purpose of activating a
TCP/IP connection to the Server located at the provider site for submitting queries. The *Dispatcher of Arriving Messages* is a module implementing a daemon that awakes with periodic cadence to verify if new messages have been stored in the mailbox. New messages are examined and the opportune manager is activated depending on their content. The *Answer Manager* and the *Answer Interface* are activated when the user wishes to have the results of previously submitted query displayed. Finally the *Manager of Global Dictionary Modification* and the *Repository Manager* are modules devoted to the management of the data repository.

The architecture of the distributor site is shown in Figure 9.
3.2. Wrappers

Wrapper functionalities are carried out by the provider site. In particular, it has to manage the set of databases associated to it in order to make their manipulation uniform, independently on languages and data models they support. Moreover, it communicates with the distributor site, accept query requests as the input and constructs query answers as the output. The architecture of the provider site is shown in Figure 10. The provider site consists of a number of software modules, whose functionalities are briefly described in the following.

The Dispatcher of Arriving Messages implements a daemon that checks the mailbox for the presence of new messages, examines newly received messages and activates the appropriate management module. The TCP/IP server implements a server daemon process which waits for the activation of a connection requested by client processes located at the distributor site. The Query Manager takes the query as the input and sends it to the DBMS Interface by preliminarily verifying its syntactic correctness. When the corresponding answer is received, the Query Manager restructures it in suitable form and sends it to the TCP/IP Server or to the Distributor Site Manager, depending on the type of connection which was established. The DBMS Interface translates queries submitted by users into the language supported by its corresponding DBMS and submits it to the DBMS itself. Thus, the overall query translation takes place in two steps. First, QBE queries are translated into SQL at the distributor site. Second, SQL queries are translated into local languages by DBMS interfaces at the provider sites. Obviously, one DBMS interface will be needed for each DBMS to be accessed. In the present release of the system, the only interfaces we actually implemented allow access to Postgres and Oracle databases. Moreover, the DBMS interface is responsible for sending query answers to the Query Manager.

Finally, the Scheme Loader and the Dictionary Manager are modules devoted to the management of Exported Scheme Dictionary.

4. The Web User Interface

NET_R interface exploits dynamic construction of HTML and JavaScript pages to supply services to the user. At present, the interface is in Italian, even if an English version will be linked soon. Main page sections included in the Web interface and their hierarchical structure are depicted in Figure 11.

The system includes an on-line help and a glossary to assist the user. After logging in, the user can choose among two pages which activate the sections for the visualization of off-line constructed query results and for repository navigation, respectively.
The former section includes a number of HTML and JavaScript pages by which the user can select one or more query results of interest and display them. We note that also result tuples displaying is embedded within an HTML and JavaScript dynamic page.

Figure 10. Architecture of Provider Site
The latter section allows the user to select a navigation method amongst the four available ones. Depending on this choice, several other pages are accessed which guide the user within the metadata towards eventually selecting the database of interest.

A further section of the interface consequently accessed allows the user to either formulate a new query through a QBE-like query interface or to request the execution of a predefined query. The Web interface includes also several other pages which supply further minor information services to the user.

Our system is accessible at the address:

http://isi-cnr.deis.unical.it:1080/~netdist

by typing the string “ursino” as the login specification. Note that, for security reasons, no “real” IGCO database has been linked to the system, whereas a couple of example databases can be freely accessed.

Figure 11. Tree structure of the Web interface

In the Figures 11, 12, 13 and 14 we show some screens displayed using the object-guided access; in particular the Figure 11 shows how the access method is
chosen whereas Figures 12, 13 and 14 show the object-guided access example described in the Section 3.

Figure 12. The Web Interface: Choosing the Access Method

Figure 13. Object-guided access: the First Level Abstract Scheme
An architecture for accessing a large number of autonomous, heterogeneous databases

Figure 14. *Object-guided access: the Second Level Abstract Scheme*

Figure 15. *Object-guided access: the Third Level abstract Scheme*
5. Conclusions

In this paper we have described an approach to the integrated access to heterogeneous distributed databases exploiting a structured global dictionary as the core of a mediator based architecture. The repository is used as a navigation tool through system metadata. Information of local databases is filtered by so called provider sites, which are in charge of performing a first level normalization of data formats whereas user functionalities are realized by the distributor site, which is in charge of supervising information distribution and presentation in the system. The system includes a graphical Web-based interface which can be accessed through any Web browser. The WWW interface exploits the information encoded in the repository descriptions to help the user of the integrated system in locating a local database of interest and in formulating queries on it. Querying is based on a simple QBE-like graphical format which is first translated to SQL and then to the local database language where the query is submitted to.

In its present version, NET_R relies on the Italian Central Governmental Offices data repository, but can run on any other data repository of suitable form.

NET_R supports the following main functionalities:

- dynamic construction of HTML and JavaScript pages for the visualization of repository schemes (including local database schemes of information provider sites);
- repository navigation for accessing underlying databases according to user's information needs;
- graphical query formulation support through a QBE-like interface in WWW format;
- management of user query submission and query result reception to/from remote provider sites (where queried database of interest are physically located);
- dynamic construction of HTML and JavaScript pages for the visualization of user query results.

Our system has been released in a prototypal version to AIPA; AIPA has been using it for some months with encouraging results.

We have presented here a centralized architecture of the distributor site (our mediator). In order to scale up the system to a greater number of data sources and users, the design of a distributor architecture for the distributor site appears to be mandatory and it is one of the topics of a next research.
An architecture for accessing a large number of autonomous, heterogeneous databases

Acknowledgments

The authors thank S. Greco and S. Donato for stimulating discussions about federated database systems. Thanks are also due to E. Iezzi, F. Locane, T. Pugliese and P. Viola for their contribution to the preliminary implementation of the NET_R system. Special thanks to Philippe Thiran for helping with translating this abstract’s paper to French.

References


