An Information Broker For Integrating Heterogeneous Hydrologic Data Sources: A Web Services Approach

Fabio Calefato[1], Attilio Colagrossi[2], Domenico Gendarmi[1], Filippo Lanubile[1], Giovanni Semeraro[1]
1 Dipartimento di Informatica, University of Bari, Via E. Orabona, 4 – 70125, Bari, Italy
{calefato,gendarmi,lanubile,semeraro}@di.uniba.it,
2 Dipartimento Tutela delle Acque Interne e Marine, APAT, Via Curtatone, 3 – 00185, Rome, Italy
attilio.colagrossi@apat.it

Abstract. Data integration systems have been developed to provide an integrated view of related data sources, so that all the information needed can be accessed uniformly, transparently and independently of the physical storage. In this paper we report our experience in developing the first prototype of an information broker for APAT, the Italian Agency for Environmental Protection and Technical Services. The broker provides integrated access to a huge amount of hydrological information stored into a collection of heterogeneous data sources.

1 Introduction

The pervasiveness of the network technology and the Internet has allowed the access to a multitude of interrelated data sources. Accessing data sources individually and then combining the results manually every time an information is needed can be awfully time-consuming. These issues are exacerbated especially in those environments where information plays a central role and has a high economic value, posing real economic threats.
Data integration systems have been developed to provide a global view of a collection of related data sources, so that all the information needed can be accessed uniformly, transparently and independently of the physical storage location, as if it was stored into a single data source [1].

The problem of integrating heterogeneous data sources has been deeply investigated in the last two decades. Previous research has shown a number of solutions, which can be broadly grouped into two main approaches, namely Data warehouse [2] and Mediator [3, 4]. The Data warehouse (or Materialized) approach provides a centralized static copy of information extracted in advance from multiple data sources [5].
Being a static copy, a data warehouse cannot be updated. Hence, this approach is not suitable for an environment where information change rapidly and required data is unknown *a priori* [6]. The Mediator (or Virtualized) approach comprehends a number of different implementations, such as the Knowledge-based Information Integration Systems [7], the Agent-based Systems [8], the Information Retrieval Systems [9] and the Federation Information Systems (FIS) [10].
FIS, in particular, grants the data sources composing the integration system to be:

- *autonomous*, as they can continue to operate independently;
- *heterogeneous*, as they can be produced by different vendors and use different data models as well as different query languages;
- *distributed*, as they can be physically located on different hosts.

In this paper we report our experience in developing the first prototype of an information broker which provides integrated access to a collection of related hydrological data sources.
The remainder of this paper is structured as follows. In Section 2 we present the project goal and its domain. In Section 3 we illustrate the integration process and the architecture of the prototype. In Section 4 we discuss some related systems. Finally, in Section 5 we draw conclusions and provide future directions of development.

2 Project Domain and Goal

The Agency for Environmental Protection and Technical Services (APAT) of Italy was established in 1999 to carry out scientific and technical activities in the national interest to protect the environment, water resources and the soil. It is subject to the supervision of the Ministry of the Environment and Territorial Protection and is integrated into the Environmental Agency System, which provides consulting, service and support to several other government agencies.
APAT is structured in several departments, which are themselves divided into smaller units, each offering intradepartmental services to the Agency. The Data Gathering and Management Service (DGMS) of the Inland and Marine Waters Protection Department gathers collections of hydrological, hydrographical and water quality data through monitoring networks covering the whole national territory. Data collected are huge and varied, as they include climatic, hydrometric, cartographic and water pollution measures. Furthermore, DGMS stores and makes available the data hydrological reports concerning all the hydrological and hydrographical phenomena observed on the whole national territory since 1921 [11].
Although all the information is owned by the same organization, the huge amount of information gathered is managed by different departments and units. Besides, given the large diversity in syntax and semantic of the data collected, measures are stored

into several independent systems, each based on the technology thought to be the most appropriate for the data type. Presently, DGMS uses and maintains five different and independent systems, built adopting different information storage technologies, namely MySQL, PostgreSQL, Tamino XML, Red Brick and AutoCAD. Furthermore each data source represents the back-end of an existing autonomous application that is currently operative and thus it cannot be dismissed.
The goal of the project is to develop an information broker to provide a full and user-transparent integration of the heterogeneous data sources maintained by DGMS, ensuring, at the same time, the existing legacy applications that operates on them to continue operating autonomously, without undergoing any sort of modification.

3 Information Broker

The storage of data concerning the hydrology, hydrography and water quality of the national territory requires technologies suitable to support large amount of data with strong heterogeneity and different needs of accessing and fruition. Hence, within this wide, varied and continually changing context we have applied the Mediator approach, thus building a Federation Information Systems to provide integrated access on demand to all the data sources.
According to the taxonomy of FIS proposed in [10] our broker is a Federated Database System (FDBS), where all the components in the federation are structured sources accessible by a specific query language. A FDBS typically provides a federated schema to ensure full location and schema transparency to its users. It has a static architecture in that a FDBS does not allow a flexible structural change management of its components. Unlike the typical FDBS implementation, our broker is also enriched with a set of web services used to enable the provision of data on demand, whilst keeping the underlying data sources unchanged and autonomous.
In the first prototype the federation is composed of four databases managed by two distinct DBMS, namely MySQL Server, used for collecting real time measures, and Tamino XML Server, used for collecting data on extreme hydrological events and hydrography of the territory.
In the next sub-sections we provide further details on the schema integration process and the layered architecture.

3.1 Schema Integration Process

Schema integration refers to the process of integrating data-source local schemas into a single federated schema that will provide a global view of the federation.
Fig. 1 shows the four-step bottom-up process completed at design time, used to develop our federated schema and similar to the one proposed in [12].
The local schemas of the different component databases within the federation represent the starting point for building a federated schema.
Due to the heterogeneity of the underlying data models, the first step in the integration process is to transform the local schemas into so-called export schemas,

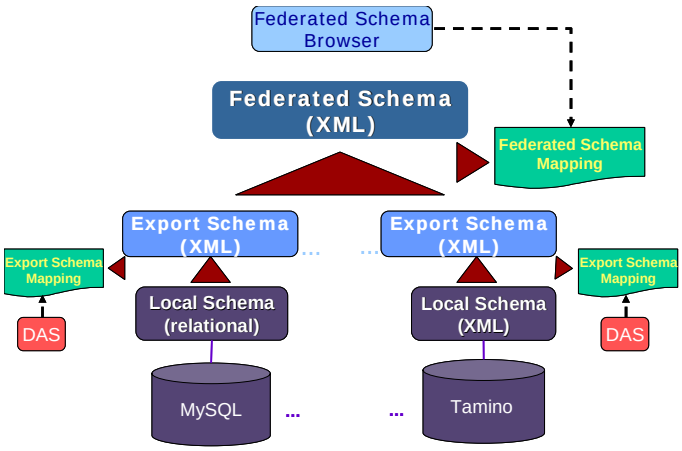

Fig. 1. Schema Integration Process

expressed in a common data model (CDM). Several options were weighted up to choose the most appropriate CDM, including relational data model, XML and OWL. Our choice was to use the XML data model because today it is the de-facto standard language to exchange information between applications and it is supported by most of the existing DBMS. Once the data model heterogeneity is overcome, the next step is the creation of the export-schema mappings, which are XML files manually generated at design time from each export schema. Such files contain the mappings between the local and export schemas, that is, the correspondences between low-level data stored and high-level domain concepts.

The third step in the integration process is the construction of the federated schema, which is supposed to represent the logical model of the virtual database containing all the data available within the federation. The federated schema is the result of the merging of all the export schemas. During the merge, all the possible conflicts have to be identified and solved. This is accomplished through two distinct activities.

The Correspondence Investigation activity searches for correspondences among the export schemas. The output of this activity is a set of conflicts, grouped in:

- *naming conflicts*, i.e. either different names are used to identify the same concept, or the same name is used to identify different concepts;
- *structural conflicts*, i.e. the same concept is represented with different structures in different schemas.

This set of conflicts is the input of the next activity, the Conflict Resolution. For naming conflicts where the same concept is identified with distinct names, a common name to use in the federated schema is chosen. For the other kind of naming conflicts the name of one of the two distinct concepts with the same name is changed. For structural conflicts, instead, each conflict is solved defining a new ad-hoc structure. Once the federated schema has been obtained, the last step in the process is to manually generate the federated-schema mapping file, an XML file that stores the correspondences between:

- complex concepts and simple concepts distributed in the different export schemas;
- simple concepts and constraints that characterize them;
- simple concepts and services able to retrieve them.

A complex concept is a concept that can be decomposed in a set of simple concepts. A simple concept, instead, is an atomic concept which cannot be further decomposed and has to be instantiated by a set of constraints.

3.2 Layered Architecture

Fig. 2 shows the broker architecture organized in three main layers:

1) the wrapper, which resolves the technical differences among the data sources available;
2) the federation, which offers a uniform way to access the data stored in data sources;
3) the presentation, which allows users to perform global queries.

3.2.1 Wrapper Layer

In the wrapper layer a Data Access Service (DAS) was developed to wrap each data source available and extract the information required on demand. Each DAS is a wrapper implemented as a web service that provides a WSDL interface to allow the remote invocation of the service via SOAP.

This solution ensures a transparent access to all distributed, heterogeneous and autonomous data sources available, since each DAS can be developed in a different implementation languages, can access sources managed by different DBMSs and can run on different platforms. However, regardless of how a DAS is programmed, it provides a unified and seamless way to access underlying information.

Each DAS provides a standard interface method to query the appropriate data source, thus the information about how actually access to the data sources is hidden inside, and the components on upper layer have a unique way to access the data. Nevertheless, even if a DAS has a uniform interface its implementation is tightly bound to the database and the DBMS it is related to.

The proposed architecture of a DAS implemented as web service grants the system to be both highly scalable, because a new data source can be added by developing a new DAS without critical effects on the upper layers, and fault tolerant, since failures in any DAS will not cause the failure of the whole broker. Furthermore, the use of web services technology and standard protocols such as HTTP, SOAP and WSDL, prevents firewall traversal issues.

Fig. 3 shows the query execution related to a MySQL database, as it is performed by a DAS. The DAS receives in input a simple concept and a set of constraints, then uses the export schema mapping file to find what query to execute. In order to actually execute the query, the DAS translates the query in SQL syntax and execute it by JDBC API.

The result of the query execution, in this example a result set, is then transformed into a XML document serialized in a string and returned to the federation layer.

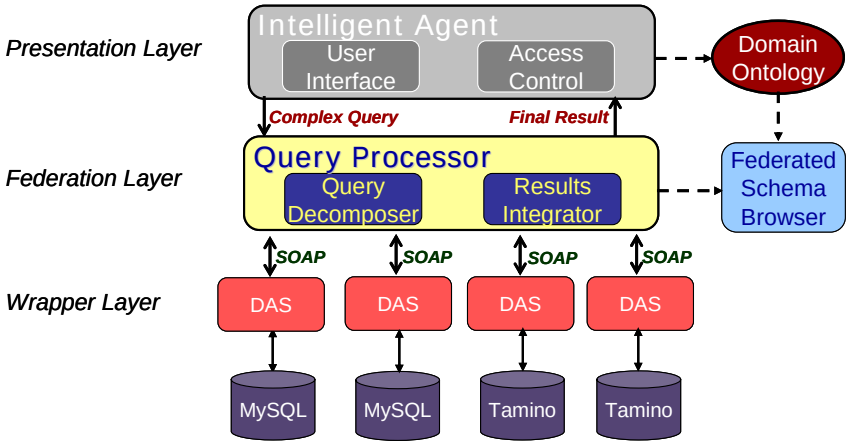

Fig. 2. Information broker architecture

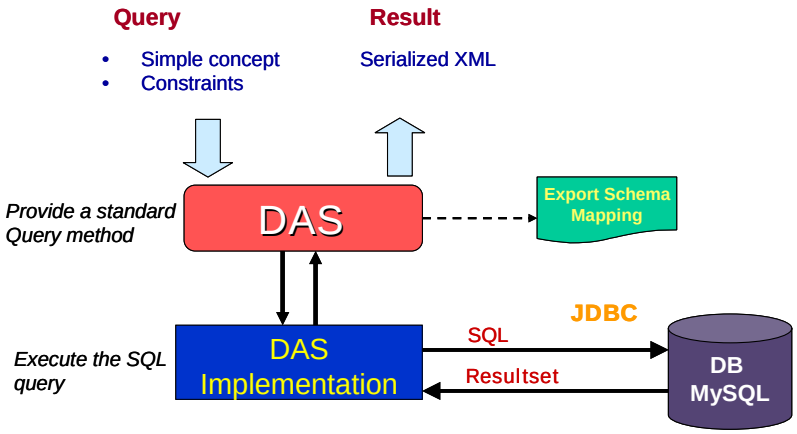

Fig. 3. Query execution performed by a DAS

3.2.2 Federation Layer

The federation layer offers a uniform and transparent access to the data sources through the Query Processor component, which is made of two subcomponents: the Query Decomposer and the Results Integrator (see Fig. 4).
The Query Decomposer takes in input the complex query expressed as a complex concept and a set of related constraints. Thus, it reduces a complex query into as many simple queries as the number of simple concepts that constitute the complex one. The Query Decomposer uses the Federated Schema Browser, which is an API built to provide high-level access to the federated schema mappings file, and find the

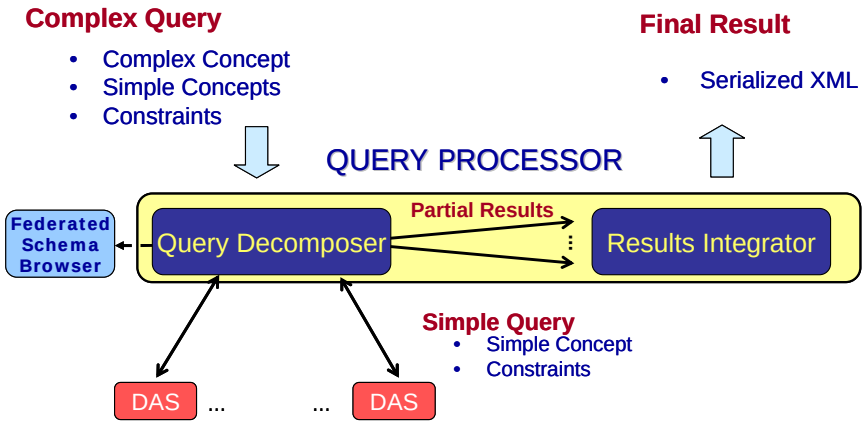

Fig. 4. Query Processor and its components

appropriate subset of constraint that characterize a simple concept. The Federated Schema Browser is used to:

- retrieve a concept, complex or simple, giving in input a name;
- retrieve all the complex concepts included in the federated schema;
- retrieve all constraints related to a given simple concept;
- find the DAS able to obtain a given simple concept.

Hence, once the Query Decomposer has created the simple queries, again it uses the Federated Schema Browser to obtain the mappings between a simple concept and the DAS capable to obtain that concept, so as to invoke the appropriate service for each concept needed.

The Query Decomposer uses the WSDL interface provided by the DAS to actually invoke the right service. All the results returned by every single DAS are passed to the Result Integrator as a set of serialized XML documents.

The Result Integrator collects the partial results obtained by each DAS queried, creates the resulting serialized document and finally returns it to the presentation layer as the final result. Such a solution preserves information about the location of the concepts, encapsulating it inside this layer and hiding it to the components in the presentation layer above. The Query Decomposer does not know how data sources have to be actually queried, because this kind of information is hidden in the wrapper layer. Thus, the QP will perform a query always in the same way, whatever the DAS to be invoked.

3.2.3 Presentation Layer

The presentation layer is composed of the Intelligent Agent and the Domain Ontology. The former represents the user interface of the system. Its main tasks are to authenticate users, driving them in the query formulation according to their adaptive profile, and to display query results. However, in the first prototype the presentation layer is just an ordinary dynamic web-based UI, which has been implemented with Java Servlet and JSP technology.

The Domain Ontology represents our hydrological domain model. An ontology is a conceptual model used to represent concepts related to a specific domain and their relationships [13]. In our architecture the Domain Ontology is meant to share knowledge about the hydrological domain and to let the Intelligent Agent browse all concepts available, thus helping it in the query formulation process.
A detailed explanation of how the agent and the ontology will be used is beyond the scope of this paper, as they will become operational only in the second prototype.

4 Related Work

In the last fifteen years a lot of research has been done in the field of the integration of heterogeneous information sources.
The Stanford-IBM Manager of Multiple Information Sources project, or TSIMMIS [14, 15], was developed by Stanford University in conjunction with IBM. A key aspect in TSIMMIS is the canonical data model, named Object Exchange Model (OEM). The OEM model forces no regularity on data and then it is possible to represent heterogeneous, changing information. The lack of a global schema in TSIMMIS, makes it possible to integrate new data sources whose schemas are partially unknown or varying during time. However, the downside of this approach is that the responsibility of resolving possible semantic heterogeneity is left to end users during query execution. Conversely, our information broker anticipates the conflict resolution at design time, during schema integration process.
The Information Manifold (IM) [16, 17] is a project developed at AT&T Bell Laboratories, with the main focus of providing declarative ways of relating a global schema to information sources, and evaluating queries based on descriptions of these information sources. IM has a global conceptual model known as a world view, against which queries are expressed. The IM approach in evaluating queries depends crucially on the information sources descriptions provided and, hence, information providers would take care to ensure that their source descriptions accurately reflect the contents and capabilities of the source relations. IM is designed using a top-down strategy, thus starting with a global information need and later adding those sources that contribute to this need. On the other hand, our broker is designed according to a bottom-up strategy, starting directly with the integration requirement of a set of data sources.
IBHIS (Integration Broker for Heterogeneous Information Sources) [18, 19] is a project funded by the UK's Engineering & Physical Sciences Research Council (EPSRC). The global view of the distributed data is achieved through the creation of one or more federated schemas, according to the user requirements. The central idea is that data become available as services on demand, without affecting the operational systems. In this context, the characterization is more data-as-a-service rather than software-as-a-service [20]. Analogously to our system, the first IBHIS prototype is based upon a FDBS enriched using web services as data source wrappers. However, in IBHIS the health and social-care domain has been chosen to simulate an integration need and all the integrated data sources are managed only by relational DBMS, namely MySQL and IBM DB2. Conversely our system grew out

of an authentic integration need. Consequently, our system had to solve the problem of integrating data sources implemented using different data models and technologies such as RDBMS (e.g., MySQL and PostgreSQL), XML-native DBMS (e.g., Tamino XML Server), and computer-aided drafting systems (e.g., AutoCAD).

5 Conclusions and Future Work

The development of an integration architecture helps build a well-factored, agile and well-integrated set of applications and services, thus balancing the requirements of the enterprise business against those of individual applications. Moreover, the development of an integration architecture helps users to focus on specifying what data they want rather than on describing how to obtain it. Users are relieved from the burden of finding the relevant data sources, interacting with each of them separately and, then, manually combining the data returned.

In this paper we have presented a project to develop an integration system which will provide an uniform view of the separate hydrological repositories managed by multiple departments of a government environmental agency.

In the next planned releases, other than accessing further data sources, the system will include an intelligent agent that will model adaptive user profiles to provide a personalized guidance in retrieving the information needed. In order to improve the query formulation process, the ontology will be enriched with additional hydrological concepts. According to the user profile the intelligent agent will extract only those ontological concepts which are relevant to the specific user. Selected concepts will then be prompted via the web user interface. Such an approach should relieve the user from looking for the right concepts which are buried in the wide amount of available data.

References

1. N. Tatbul, O. Karpenko, C. Convey, Data Integration Services, Technical Report, Brown University, Computer Science, 2001.
2. J. Widom, Research Problems in Data Warehousing, in: Proceedings of the 4th Conference on Information and Knowledge Management (CIKM, 1995).
3. G. Wiederhold, Mediators in the Architecture of Future Information Systems, *IEEE Computer* **25**(3), 38-49 (1992).
4. G. Wiederhold, Interoperation, mediation, and ontologies, in: Proceedings of the Workshop on Heterogeneous Cooperative Knowledge Bases, International Symposium on Fifth Generation Computer Systems (FGCS, 1994), pp 33–48.
5. P. Dorian, *Business Modeling and Data Mining* (The Morgan Kaufmann Series in Data Management Systems, 2003).
6. V. Raman, I. Narang, C. Crone, L. Haas, S. Malaika, T. Mukai, D. Wolfson, and C. Baru, Data Access and Management Services on Grid, in: The Fifth Global Grid Forum (GGF5, 2002).

7. N.W. Paton, C.A. Goble, S. Bechhofer, Knowledge Based Information Integration Systems, *Information and Software Technology* **42**(5), 299-312 (2000).
8. W.Shen, D.H. Norrie, Agent-Based Systems for intelligent Manufacturing: A State-of-the-Art Survey, In *International Journal of Knowledge and Information Systems*, **1**(2), 129-156 (1999).
9. V. Gudivada, V. Raghavan, W. Grosky, R. Kasanagottu, Information Retrieval on the World Wide Web, *IEEE Internet Computing* **1**(5), 58-68 (1997).
10. S. Busse, R. Kutsche, U. Leser, H. Weber, Federated Information Systems: Concepts, Terminology and Architectures, Technical Report Forschungsberichte des Fachbereichs Informatik 99-9, Technische Universitat Berlin, 1999.
11. A. Colagrossi, G. Nardone, Technologies For Storing, Processing and Fruition of Hydrological Data, in: 2nd International Conference on River Basin Management (WIT Press, 2003).
12. A.P. Sheth, J.A. Larson, Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases, *ACM Computing Surveys* **22**(3), 183-236 (1990).
13. T. Gruber, Towards principles for the design of ontologies used for knowledge sharing, *International Journal of Human-Computer Studies* **43**(5/6), (1995).
14. H. Garcia-Molina, Y. Papakonstantinou, D. Quass, A. Rajaraman, Y. Sagiv, J. Ullman, V. Vassalos, and J. Widom, The TSIMMIS approach to mediation: data models and languages, *Journal of Intelligent Information Systems* **8**(2), 117-132 (1997).
15. Y. Papakonstatniou, H. Garcia-Molina, J. Ullman, MedMaker: A mediation system based on declarative specifications, in: Proceedings of the International Conference of Data Engineering, (ICDE, 1996), pp. 132-141.
16. A.Y. Levy, D. Srivastava, T. Kirk, Data model and query evaluation in global information systems, *Journal of Intelligent Information Systems* **5**(2), 121-143 (1995).
17. A.Y. Levy, A. Rajaraman, J.J. Ordille, Querying heterogeneous information sources using source descriptions, in: 22nd Very Large Data Bases Conference (VLDB, 1996).
18. I. Kotsiopoulos, J. Keane, M. Turner, P. Layzell, F. Zhu, IBHIS: Integration Broker for Heterogeneous Information Sources, in: 27th Annual International Computer Software and Applications Conference (COMPSAC, 2003).
19. M. Turner, F. Zhu, I. Kotsiopoulos, M. Russell, D. Budgen, K.H. Bennett, P. Brereton, J. Keane, P. Layzell, and M. Rigby, Using Web Services Technologies to create an Information Broker: An Experience Report, in: 26th International Conference on Software Engineering (ICSE, 2004).
20. K.H. Bennett, P. Layzell, D. Budgen, P. Brereton, L. Macaulay, M. Munro, Service-Based Software: The Future for Flexible Software, in: 7th Asia Pacific Software Engineering Conference (IEEE Computer Society Press, 2000).