A Reference Architecture for Integrating Heterogeneous Information Sources using XML and Agent model

Ngamnij Arch-int[1]
Peraphon Sophatsathit[2]
Advanced Virtual and Intelligent Computing (AVIC) Research Center
Department of Mathematics, Faculty of Science, Chulalongkorn University, Thailand
ngamnij@kku.ac.th[1], Peraphon.S@chula.ac.th[2]

Abstract

Consolidation of data from different repositories has always been one of the most challenging problems for researchers and implementers alike. Many ad-hoc solutions have been devised to cope with this predicament, ranging from conventional approaches such as format conversion, open-connectivity base, distributed processing, to new paradigms such as Object Exchange Model, tools, content language protocol, and description-logic based system, etc. This paper proposes a layered-architecture that separates various stages of information exchange idiosyncrasies from one another. Heterogeneity is consequently transparent to the users. The advantages resulting from the application of existing tools and techniques are, to some extent, offset by the inherent operational overhead. The benefits of having plug-and-play stacks, coupled with the flexibility of standard user-level application such as XML, will eventually encompass a wide range of heterogeneous information computing and communication.

1 Introduction

Many organizations rush to advertise their business and operations on WWW. Each organization designs and develops their information independently, catering to the needs of their own departments. As a consequence, various types of information sources within an organization, ranging from *structured*, *semi-structured*, and *unstructured information sources,* typically reside on different physical hosts, operating systems, or DBMSs. We call these different information sources the *Heterogeneous Information Sources (HIS)*. The structured information sources are usually constructed from predefined schemas; the semi-structured information sources, in most cases, are constructed from similar configuration, but in weaker form of predefined schemas; yet the unstructured information sources are constructed without any restrictions. Examples of such information sources are structured database files, semi-structured XML (Extensible Markup Language) documents, and unstructured plain text, respectively.

The inherent variations in data definitions imposed by each host pose a great challenge for integrating efforts on distributed processing at each site. Two predominant problems arising from heterogeneity of data are *schematic heterogeneity* and *semantic heterogeneity*. Schematic heterogeneity results from different local schemas that govern the underlying data model [1]. Consequently, applications of cross-platform exchange inevitably induce non-conforming integration. Semantic heterogeneity, on the other hand, occurs when there is a disagreement about the meaning, interpretation, or intended use of the same or related data [2] (e.g., name conflict, scale conflict [1, 3]). A number of works have been carried out to overcome these problems, for example, TSIMMIS [4], SIMS [5], IM [6], COIL [7] and SHOE [8]. Regardless of the principles applied to integrating data from HIS in Web-based environment, none of these approaches support the entire *flexibility*, *robustness*, *scalability*, *interoperability* and *portability properties.*

This paper proposes a reference architecture that serves as a unified framework for accessing HIS in Web-based environment via the Agent Model. The paper will focus on integrating data from HIS, and resolving the heterogeneity of data by proposing a unified XML-based data environment. A particular system developed and deployed based on the reference architecture will provide flexibility, robustness, scalability, interoperability, and portability properties. Such extensive supports render the target environment a transparent homogeneous Web-based repository. The language will also be used as a representation basis for mobile agent communication within the target environment.

The paper is organized as follows. Section 2 furnishes an overview and details of the reference architecture, as well as the underlying agent model architecture, that is essential for the integration of heterogeneous information sources. An example of leveraging XML to resolve the heterogeneity of data based on the proposed reference architecture is also illustrated. Section 3 concludes this research effort along with suggested future work.

2 Reference Architecture

2.1 An Overview

The proposed reference architecture, depicted in Figure 1, consists of four main layers, namely, Presentation, Mediator, Search, and Resource Layers. The main objective of employing layered-architecture is to support component *portability* which enables a system to be run on a variety of platforms. Each layer is described briefly, as follows:

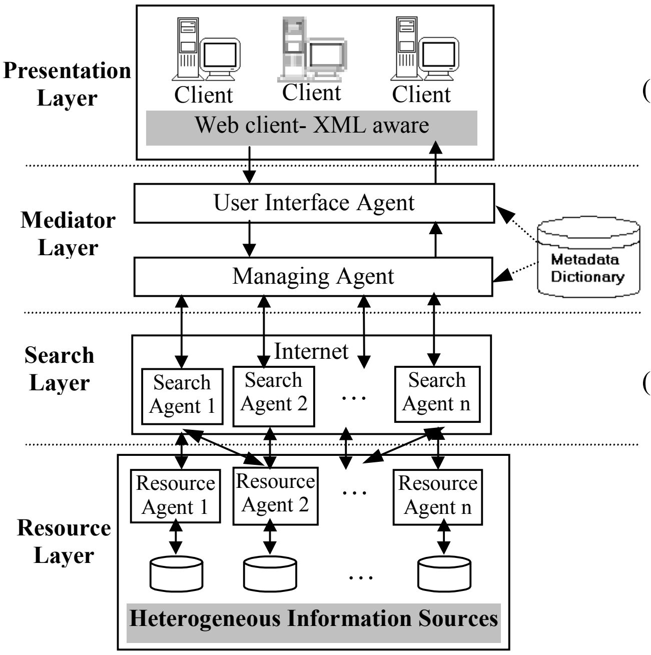


Figure 1 A reference architecture for integrating heterogeneous information sources on WWW

(1) **Presentation Layer** is the highest layer of the reference architecture consisting of the *Web client component*. This layer is designed specifically to enable direct user computing through graphical user interface in Web environment. The Web client component communicates with the User Interface Agent of the Mediator Layer by forwarding users' requests and receiving the returned results from the User Interface Agent in XML format. In doing so, cross systems *interoperability* can be achieved with XML flexible data model;

(2) **Mediator Layer** is the core layer of the proposed reference architecture. This layer consists of three principal components, namely, the *Metadata Dictionary*, the *User Interface Agent*, and the *Managing Agent*. The primary responsibility of this layer is to bridge the gaps between format-specific components at the lower layers and the presentation layer. Data integration is designed based on the notion of "*Dynamic Integration*", i.e., there is no provision for pre-defined schemas or global schemas to restrict user's operations [3]. A virtual schema encircling various information sources stored in the Metadata Dictionary is furnished for the users to create their own customed views and query over interest schemas. A query posed by a user is statically decomposed into sub-transactions corresponding to the description of the designated local information sources. This technique supports *scalability* that permits adding or dropping new information sources without affecting the overall system configuration. Thus, heterogeneity problems can be resolved at query time for local and remote processing;

(3) **Search Layer** consists of *Search Agents* which are mobile agents. Each Search Agent is responsible for carrying a sub-transaction and related user's information to a destination source through the Resource Layer. The Search Agent also communicates with the Resource Agent to collect and ship the result obtained from the Resource Agent back to the Mediator Layer; and

(4) **Resource Layer** encompasses *HIS* and *Resource Agent* which are stationary agents and often referred to as agent hosts. Each Resource Agent connects directly with a local information source. Its responsibility is to serve a request arriving from the Search Agent and transform the returned results into canonical data model represented in XML format. The returned results are then packed and sent to the Search Agent for shipping back to the Managing Agent.

The layered-architecture of the reference architecture provides greater *flexibility* and *robustness* in that agent mechanism can be employed as a means to send and receive information between layers, whereby the interaction patterns among agents can be altered without modifying agent code, as long as the protocol remains unchanged. Robustness offers mobile agents to operate autonomously and asynchronously within and across layers, thus enabling the system to sustain the effects of intermittent network operation or failure in information sources. Details of each layer and component are described in sections below.

2.2 Presentation Layer

The Web client component provides location and heterogeneous transparencies to the users for maximal flexibility. The users can freely maneuver their search requests without any priori knowledge of local schemas, query languages, and data models of the information sources. The responsibilities of this layer are (1) support graphical user interface for users to view the entire schemas of various information sources that are consolidated by the User Interface

Agent; (2) provide a unified-query form for the users to create their customed views and pose query against their views; (3) formulate and validate users' requests to insure that correct information is passed to the User Interface Agent; and (4) display the returned results from the User Interface Agent in XML format.

2.3 Mediator Layer

The components of this layer are described below.

2.3.1 Metadata Dictionary

The Metadata Dictionary stores all data descriptions and associated information of various information sources that are used by the User Interface Agent and the Managing Agent. The Metadata Dictionary supports user-level transactions performed by the User Interface Layer and the Management Layer as follows:

(1) **Generating global transaction** which provides the virtual data for user's request validation and global transaction creation;
(2) **Privilege analysis** which provides user's access privilege for the information sources;
(3) **Optimizing transaction** which provides information about data allocation, local access path and statistical information needed for creating query;
(4) **Translating global transaction** which provides information mapping between virtual data and physical data for sub-transaction creation; and
(5) **Result integration** which provides the virtual data corresponding to the user's requirements, whereby eliminating the heterogeneity problems.

The scope of the Metadata Dictionary can be categorized as follows:

(1) **Information source descriptions**

- Data descriptions and constraints of the data stored in various information sources,
- The relationships between data descriptions and the corresponding information source names,
- The relationships between virtual attributes/ entities and actual attributes/entities of the information sources,
- Data model for each information source, and
- The domain name, host name, physical location of each information source, and the configuration of each information source that are necessary for the Search Agent.

(2) **The privilege information**

- Users access privilege of individual information source, and
- The authenticated credential information needed by the Search Agents.

(3) **The optimization information**

- Fragmentation and replication information needed by Transaction Optimizer to generate sub-transactions, and
- The statistical information for evaluating the cardinality of query intermediate results that are needed for the query plan.

2.3.2 The User Interface Agent

The User Interface Agent is a stationary agent responsible for displaying the overall schemas of various information sources to users, generating global transactions associated with the user's requests, as well as validating the syntax by means of the Metadata Dictionary. A "*global transaction*" is a visual user requirement represented in standard SQL format using virtual attributes and relations that may encompass one or more physical information sources. The user's information (e.g., name, address, etc.), type of a user (e.g., local or remote), and a global transaction would be packed into a user's package before forwarding to the Managing Agent. The User Interface Agent subsequently forwards the results obtained from the Managing Agent to the upper layer Web clients.

2.3.3 The Managing Agent

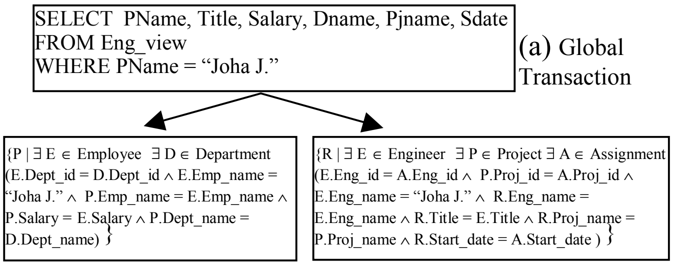
The Managing Agent is a stationary agent responsible for generating and transmitting sub-transactions through the Search Agents, as well as integrating the results returned by the Search Agents into the Unified XML-based data, which will be referred below. The Managing Agent is made up of four main modules, which are as follows:

(1) *Privilege Analysis* determines user's privilege to access the information sources based on relevant data descriptions defined in the Metadata Dictionary;

(2) *Transaction Optimizer* consists of two components:

(2.1) *Transaction planner* generates all possible query trees based on the global transaction by acquiring information from the Metadata Dictionary, e.g., data allocation, access path, statistic information needed to determine the optimal query tree for execution,

(2.2) Execution Planner decomposes an optimal query tree into several sub-transactions and assigns each sub-transaction to the corresponding information source. All references of virtual attributes and relations in global transaction are replaced with actual attributes and relations of the destination sources for each sub-transaction with the help of the Metadata Dictionary. Figure 2 illustrates an example of the decomposition of the global transaction created by the User Interface Agent into sub-transactions.



```
SELECT  PName, Title, Salary, Dname, Pjname, Sdate
FROM Eng_view
WHERE PName = “Joha J.”
```

(a) Global Transaction

{P | ∃ E ∈ Employee ∃ D ∈ Department (E.Dept_id = D.Dept_id ∧ E.Emp_name = “Joha J.” ∧ P.Emp_name = E.Emp_name ∧ P.Salary = E.Salary ∧ P.Dept_name = D.Dept_name) }

{R | ∃ E ∈ Engineer ∃ P ∈ Project ∃ A ∈ Assignment (E.Eng_id = A.Eng_id ∧ P.Proj_id = A.Proj_id ∧ E.Eng_name = “Joha J.” ∧ R.Eng_name = E.Eng_name ∧ R.Title = E.Title ∧ R.Proj_name = P.Proj_name ∧ R.Start_date = A.Start_date) }

(b) Sub-transaction sending to Personnel information source

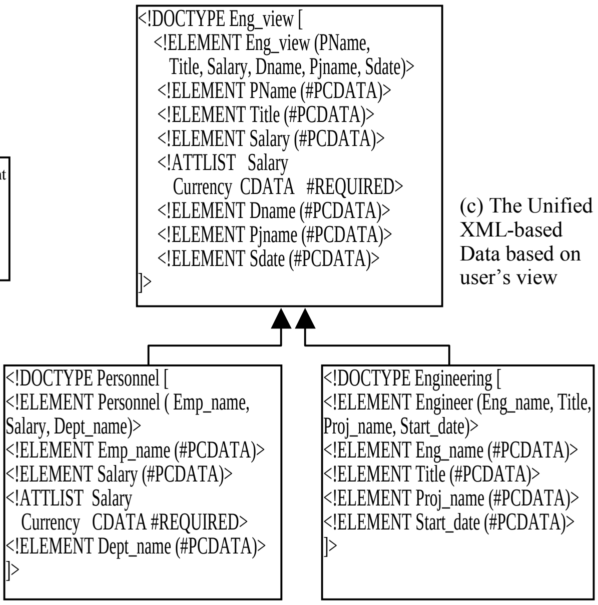
(c) Sub-transaction sending to Engineering information source

Figure 2 Decomposing the global transaction into sub-transactions associated to the local information sources

(3) *Search Agent Creator* initializes a number of Search Agents according to the number of sub-transactions to operate at remote sources. In addition, the Search Agent Creator encodes and packs all necessary information through the serialization process into each Search Agent before shipping all Search Agents to their destination sources. The information are:

- User’s privilege to information sources,
- Sub-transaction corresponding to the specific information sources,
- The address of the destination sources, and
- The authentication information of Search Agent to be trusted by agent host at the destination sources; and

(4) *Results Integrator* is designed on the notion of *Virtual Integration architectures* that temporarily help materialize query results at the time the query is posed [1]. As such, data are not replicated and are guaranteed to be fresh at query time [9]. The main responsibilities include decoding and integrating the multiple results delivered by the Search Agents into the “*Unified XML-based Data*” depicted in Figure 3 (c). Figure 3 (a) and (b) illustrates the returned results (shown only XML DTD) from different local information sources in XML format. Inconsistency and redundancy problems are thus eliminated through this combined result. The result integrating process is depicted in Figure 4.



```
<!DOCTYPE Eng_view [
  <!ELEMENT Eng_view (PName,
     Title, Salary, Dname, Pjname, Sdate)>
  <!ELEMENT PName (#PCDATA)>
  <!ELEMENT Title (#PCDATA)>
  <!ELEMENT Salary (#PCDATA)>
  <!ATTLIST  Salary
     Currency  CDATA  #REQUIRED>
  <!ELEMENT Dname (#PCDATA)>
  <!ELEMENT Pjname (#PCDATA)>
  <!ELEMENT Sdate (#PCDATA)>
]>
```

(c) The Unified XML-based Data based on user’s view

```
<!DOCTYPE Personnel [
<!ELEMENT Personnel ( Emp_name,
Salary, Dept_name)>
<!ELEMENT Emp_name (#PCDATA)>
<!ELEMENT Salary (#PCDATA)>
<!ATTLIST  Salary
   Currency  CDATA #REQUIRED>
<!ELEMENT Dept_name (#PCDATA)>
]>
```

```
<!DOCTYPE Engineering [
<!ELEMENT Engineer (Eng_name, Title,
Proj_name, Start_date)>
<!ELEMENT Eng_name (#PCDATA)>
<!ELEMENT Title (#PCDATA)>
<!ELEMENT Proj_name (#PCDATA)>
<!ELEMENT Start_date (#PCDATA)>
]>
```

(a) The XML DTD result generated from Personnel information source

(b) The XML DTD result generated from Engineering information source

Figure 3 Combining the results generated from the two sites into the Unified XML-based Data

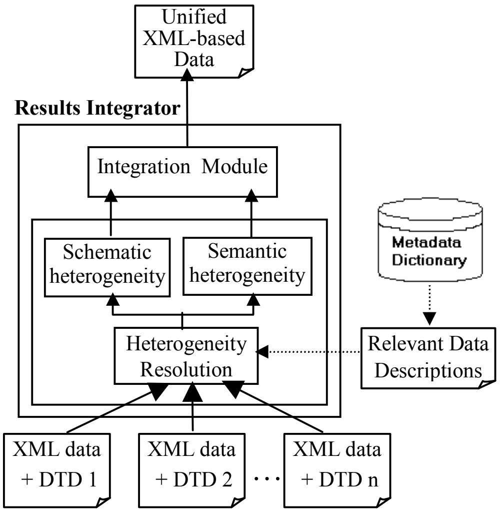


Figure 4 The Internal process of Result Integrator Module

2.4 Search Layer

The main responsibilities of Search Agents in this layer are as follows: (1) carry sub-transaction and related user’s information sent from the Managing Agent to the destination source, (2) communicate with the Resource Agent to acquire the requested information, (3) communicate with other Search Agents to exchange intermediate data among themselves, thus avoiding unnecessary data transmissions between the Search Agents and the Managing Agent, (4) pack and encode, through a serialization process, the results obtained from the Resource Agents before dispatching back to the originating Managing Agent, and (5) collect and ship the results obtained from the Resource Agents back to the Managing Agent.

The underlying search algorithm for each Search Agent can either be statically defined or dynamically loaded during its creation. We envision, however, that the static approach offers a simple yet efficient approach to search scheme without loss of operational flexibility. The dynamic approach, on the other hand, is difficult to implement across many platforms.

2.5 Resource Layer

Each Resource Agent carries out the following functions: (1) receives and unpacks the request from the Search Agent, (2) verifies the authentication credentials of the Search Agent and authorizes checking of user access privilege to the local information source, (3) establishes the connection when activated with the information source via middleware mechanisms such as ODBC/JDBC, HTTPD, C++ Interface, and (4) encapsulates in/out parameters passing between the Search Agent and the heterogeneous data via the *interface wrapper*. The interface wrapper, in turn, converts the incoming sub-transaction to appropriate data manipulation language, as well as transforms the results obtained from the execution of each sub-transaction into canonical XML format (XML data and DTD) before decoding and passing on to the Search Agent.

3 Conclusion and Future Work

Data complexity stored in different formats and sources renders integration as one of the challenging tasks to overcome. The main objective of the proposed reference architecture is to solve the heterogeneity of data by proposing a unified XML-based data environment based on layered-architecture that supports flexibility, scalability, robustness, interoperability and portability. With the salient characteristics of Agents and XML technologies, we combine these technologies in our reference architecture to resolve the heterogeneity problems of Heterogeneous Information Sources. Flexibility of the proposed reference architecture to the WWW applications is extensive but with some penalties involved. The overhead incurred by format conversion to the Unified XML-based Data form is the major hurdle to be reckoned with. We envision that through the use of such standard protocol stacks, our proposed framework should eventually find a common ground suitable for most information exchange, as the forerunner TCP/IP have accomplished. Restrictions of the proposed system are placed on preliminary query formulation stage but should enhance the update capability of the system in future work.

References

[1] Susanne Busse, Ralf-Detlef Kutsche, Ulf Leser and Herbert Weber. Federated Information Systems: Concepts, Terminology and Architectures. Forschungsberichte des Fachbereichs Informatik Nr. 99-9, 1999.

[2] Amit P. Sheth, James A. Larson. Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. ACM Computing Surveys, Vol. 22, No. 3, September 1990.

[3] Rex Jakobovits. Integrating Autonomous Heterogeneous Information Sources. Univ. of Washington Technical Report, UW-CSE-971205, July 1997.

[4] Sudarshan Chuwathe, Hector Garcia-Molina, Joachim Hammer, Kelly Ireland, Yannis Papakonstantinou, Jeffrey Ullman, Jennifer Widom. The TSIMMIS Project: Integration of Heterogeneous Information Sources. Proceedings of IPSJ, Tokyo, Japan, October 1994.

[5] Craig A. Knoblock and Jose Luis Ambite. Agents for Information Gathering. In J. Bradshaw, editor, Software Agents, AAAI Press/The MIT Press, 1997.

[6] Alon Y. Levy, Anand Rajaraman, Joann J. Ordille. Querying Heterogeneous Information Sources Using Source Descriptions. Proceedings of the 22th VLDB Conference, Bombay, India, 1996.

[7] Christian Och, Roger King and Richard Osborne. Integrating Heterogeneous Data Sources using the COIL Mediator Definition Language. SAC'00 March 19-21 Como, Italy, ACM 2000.

[8] Jeff Heflin, James Hendler, and Sean Luke. SHOE: A Knowledge Representation Language for Internet Applications. Technical CS-TR-4078, Institute for Advanced Computer Studies, University of Maryland, 1999.

[9] Daniela Florescu, Alon Levy and Alberto Mendelzon. Database Techniques for the World-Wide Web: A Survey. ACM SIGMOD Record, 27(3), 1998.