The Design of a Grid-enabled Information Integration System Based on Mediator/Wrapper Architectures

Jihwan Song[1] Sanghyun Yoo[1] Chang-Sup Park[2] Dong-Hoon Choi[3] Yoon-Joon Lee[1]

[1]Department of Electrical Engineering and Computer Science
Korea Advanced Institute of Science and Technology
Daejeon, 305-701, South Korea

[2]Department of Internet Information Engineering
The University of Suwon
Hwaseong-si, 445-743, South Korea

[3]National Technical Information System (NTIS) Division
Korea Institute of Science and Technology Information
Daejeon, 305-333, South Korea

Abstract - *The advance of Grid computing allows people not only to solve huge scientific problems using distributed computing resources but also to access heterogeneous and distributed data sources through the standard interface of Grid computing. By adopting the merits of Grid computing, traditional mediator/wrapper-based Information Integration Systems are evolving into Grid-enabled systems such as OGSA-DAI, GDMS, OGSA-DQP, etc. However, these systems have difficulties in the dynamic virtual organization environment because their administrators must create mediation schemas integrating all data sources that have a possibility to be added before they are executed. In this paper, we design a Grid-enabled Information Integration System based on mediator/wrapper, called Dynamic Mediation Service Management System (DMSMS). DMSMS functions not only to integrate data sources, but also to find them requested by users, create mediation schemas on demand, and notify data changes to the subscribers. We also point out several technical issues for developing DMSMS to be applied to National Technical Information System (NTIS). Now, we are on the way of its implementation.*

Keywords: Grid computing, Information Integration System, Mediator, and Wrapper

1 Introduction

Traditional distributed computing environments have been constructed using several existing communication techniques such as socket programming, RPC [1], Java RMI [2], DCOM [3], CORBA [4], and so on. If different communication techniques are used in these environments, the communication is difficult to be done and needs adapters for interpreting messages sent and received between constituent systems. It is too labor-intensive to develop adapters.

The need for an open standard that encourages interoperability between distributed and heterogeneous systems was a motivation for the Open Grid Services Architecture (OGSA) [5] proposed by Global Grid Forum (GGF). It combines Grid computing [6] and Web service technologies, and is developed as a new standard of distributed computing environments.

Grid computing has influenced database systems. The GGF Database Access and Integration Service (DAIS) Working Group [7] developed a specification [8] for Grid database services. The first reference implementation of this specification, OGSA-DAI [9], is a middleware product which can support the exposure of data resources onto Grids, such as relational databases, XML databases, and file systems. It provides various interfaces for popular database systems and a simple toolkit for developing client applications. The software also includes a collection of components for querying, transforming and delivering data in different ways. It was designed to be extendable so that users can add their own functionalities to the collection. Grid-enabled systems can easily access database systems wrapped by OGSA-DAI.

Information Integration Systems [10, 11] are needed by users who want data sources to be shown as a unified view. Recently, many Grid-enabled Information Integration Systems based on mediator/wrapper have been developed by exploiting OGSA-DAI, such as GDMS [12, 13] and OGSA-DQP [14, 15, 16]. However, they do not fit in the dynamic environment because their mapping schemas and rules must be predefined by system administrators before they are executed. For example, configuration files for describing the location of data sources and access methods must be created in the initialization step in OGSA-DQP.

In this paper, we design *Dynamic Mediation Service Management System* (DMSMS) that is another Grid-

enabled Information Integration System based on mediator/wrapper, which is a component of *National Technical Information System* (NTIS) [17] supported by *Ministry of Science and Technology* (MOST) [18]. Its main function is to integrate data sources in the Grid computing environment. Unlike the previous systems, however, DMSMS provides facilities to find data sources described in *Data Source Description Language* and create mediation schemas on demand using *Mediation Schema Designer*.

The rest of the paper is organized as follows. We discuss related work about the Grid-enabled Data Integration Systems in Section 2. In Section 3 and 4, we describe the design of our information integration system that is appropriate for the dynamic environment and suggest several technical issues that occur when our system is implemented. We conclude in Section 5.

2 Related Work

There are many research groups studying Grid-enabled Data Integration Systems. As results of the research, many products such as OGSA-DAI, GDMS, and OGSA-DQP have been proposed. We explain details of these systems as follows.

OGSA-DAI is a reference implementation of the specification proposed by DAIS-WG. It is a middleware product that allows various data resources including relational databases, XML databases, and file systems to be accessed via Web Services. The list of supported databases and file systems are shown in Table 1. OGSA-DAI allow data to be queried, updated, transformed, and delivered. They are deployed in a Grid environment and used to make users' data resources Grid-enabled. As a wrapper, however, OGSA-DAI just provides unified interfaces to applications (especially mediators) for integrating various data sources rather than a virtual data source integrated from heterogeneous data sources.

Table 1. Data Resources supported by OGSA-DAI

Data Resource		Version
RDBMS	MySQL	3.2.3 or above
	IBM DB2	-
	Microsoft SQL Server	-
	Oracle	8/9i
	PostgreSQL	-
XML DBMS	Apache Xindice	1.0
File Systems	Unix FS	-
	Windows FS	-

Table 2. Comparison of the characteristics of Grid-enabled Information Integration Systems

	OGSA-DAI	GDMS	OGSA-DQP
Integrating data sources	X	O	O
Resolving heterogeneity	X	O	X
Searching data sources on the fly	X	X	X

Grid Data Mediation Service (GDMS) is a component of *Grid Miner* [19] that is a research project from Vienna University. GDMS modifies OGSA-DAI to allow virtual data sources integrated from various data sources to be the data sources of OGSA-DAI. GDMS adds transformation functions to resolve the heterogeneity of distributed data sources. It also provides integrated virtual data sources using predefined mapping schemas and rules, and has a distributed query processor to execute user queries over the virtual mediation schema. However, GDMS does not support the facilities to search and integrate data sources at run-time. System administrators should select data sources and define mapping schema and rules to integrate heterogeneous data sources before GDMS is executed.

OGSA-DQP (Distributed Query Processing) is a part of a project called *myGrid* [20] that was carried out by the Manchester and New Castle University. It is a service-based distributed query processor using OGSA-DAI. OGSA-DQP decomposes a user query into local queries by traditional distributed query processing techniques, executes the local queries on the distributed nodes powered by Grid computing, composes the results sent from each local site, and returns a final result to the user. A distinguishing feature of OGSA-DQP is to use idle resources using Grid computing. This implies that OGSA-DQP divides user query into sub-queries that can be executed independently and concurrently, and the sub-queries are executed on idle resources to improve performance. However, OGSA-DQP does not provide a complete solution to the heterogeneity of distributed data sources. It just treats distributed data sources as relations in a system rather than creates virtual data sources.

We have discussed features of three representative Grid-enabled Information Integration Systems: OGSA-DAI, GDMS, and OGSA-DQP. Table 2 briefly summarize and compare their characteristics categorized into *integrating data sources*, *resolving heterogeneity*, and *searching data sources on the fly*.

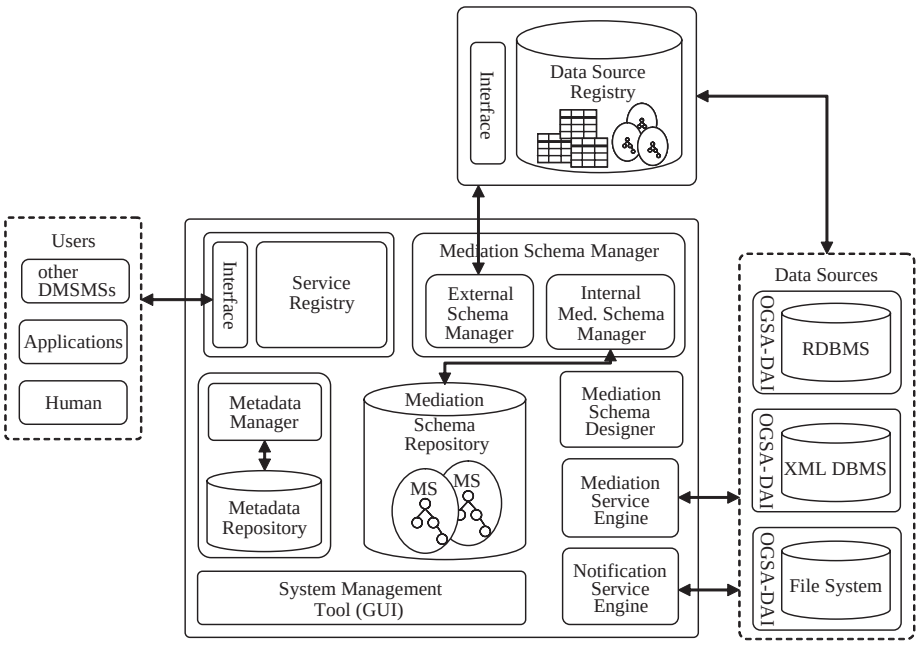


Figure 1. Architecture of DBMS

3 Dynamic Mediation Service Management System

We propose *Dynamic Mediation Service Management System* that provides the facilities for searching data sources and creating mediation schemas on demand, as well as mediation and notification services based on the mediation schema. In this section, we discuss a lack of the systems mentioned in Section 2, describe the advantages of our system, and present the architecture of DMSMS and the scenarios of *Mediation* and *Notification Services*.

3.1 Motivation and Goal of DMSMS

The previous Grid-enabled Information Integration Systems based on mediator/wrapper described in Section 2 have limitations in environments where many data sources are added and removed frequently, such as the dynamic virtual organization environment. These systems require mediation schemas at the initialization step. It is inappropriate because administrators must create mediation schemas integrating all data sources that have a possibility to be added before systems are executed.

In the dynamic environment, a Grid-enabled Data Integration System based on mediator/wrapper should provide the facilities for searching data sources from the registry that manages descriptions of data sources. It also should provide the GUI tools to support easy and convenient design of mediation schemas.

The advantages of our DMSMS are as follows: (i) Users of DMSMS can easily find data sources they need using *Data Source Description Language* (DSDL). DSDL describes data sources in detail such as *"what data sources are provided," "what the schemas of data sources mean," "where data sources are,"* and so on. (ii) DMSMS provides GUI-based *Mediation Schema Designer* tool for users to easily create mediation schemas. (iii) It provides *Notification Services* to deal with the changes of data sources in an active manner.

3.2 Architecture of DMSMS

DMSMS directly searches data sources needed by users from the external registry that stores the descriptions of data sources, provides a *Mediation Schema Designer* tool for easy creation of mediation schemas, and performs *Mediation* and *Notification Services*. In order to provide these functionalities, DMSMS consists of four components: *Main System*, *Users*, *Data Source Registry*, and *Data Sources* (see Figure 1). Below, we briefly describe these main components.

- *Main System*: This is a core part of DMSMS. It consists of *Service Registry*, *Mediation Schema Manager and Repository*, *Mediation Schema Designer*, *Mediation and Notification Service Engine*, *Metadata Manager and Repository*, and *System Management Tool*. *Service Registry* provides an interface for users to search the services of DMSMS using Web services or HTML. *Mediation Schema Manager* (MSM) provides the facility for searching data source schemas and mediation schemas, and *Mediation Schema Repository* stores mediation schemas generated by MSM. *Mediation Schema Designer* (MSD) provides a GUI tool for creating mediation schemas at run-time. Figure 2 shows the example of a mediation schema generated

by MSD. When a user queries on a mediation schema, *Mediation Service Engine* decomposes the query into local queries, sends each local query to each data source, gathers and combines results, and returns the combined result to the user. *Notification Service Engine* notifies the changes of data sources to relevant subscribers. *Metadata Manager* and *Metadata Repository* stores and manages configurations used in the system.

- *Users*: They are the subjects of using *Main System*. They can be applications, human, or other DMSMSs. While applications and other DMSMSs access *Main System* through the interfaces of Web services, human can access it using Web browsers.

- *Data Source Registry*: Data sources register their descriptions with this component. Users can find data sources here through the services of the *Main System*.

- *Data Sources*: RDBMSs, XML DBMSs, and file systems wrapped by OGSA-DAI can be data sources (refer to Table 1).

3.3 Mediation and Notification Service Scenarios

In this section, we describe *Mediation* and *Notification Services*. *Mediation Service* allows users to access more than one data sources over a mediation schema and retrieve integrated data they want. By *Notification Service*, the changes of virtual data sources are delivered to proper subscribers. Figure 3 shows the conceptual sequence diagrams of *Mediation* and *Notification Services*.

Figure 3 (a) presents the execution sequence of *Mediation Service*. It consists of the following phases:

1. A user finds data sources or virtual sources (mediation schemas) from *Data Source Registry* or *Mediation Schema Repository* using *Mediation Schema Manager* (MSM). MSM returns the descriptions of data or virtual sources needed by the user.

2. The user creates a mediation schema using *Mediation Schema Designer* (MSD) if they failed to find any appropriate mediation schema in phase 1. At this time, MSD obtains information for generating mediation schema from data sources. *Mediation Schema Manager* stores the mediation schema into *Mediation Schema Repository* and returns *Mediation Schema Reference* (MSR) to the user.

3. The user sends a query with MSR to *Mediation Service Engine* (MSE). MSE retrieves a mediation schema from MSM. It parses, decomposes, optimizes the user query, and generates local queries. Then, it sends the local queries to data sources and retrieves results from data sources. Finally, MSE transforms the results into a user's favorite form and delivers the final result to the user.

Figure 3 (b) presents the execution sequence of *Notification Service*. Their phases are as follows:

1. This phase is equal to the phase 1 of *Mediation Service*.

2. This phase is equal to the phase 2 of *Mediation Service*. (Phase 1 and 2 are omitted in Figure 3 (b))

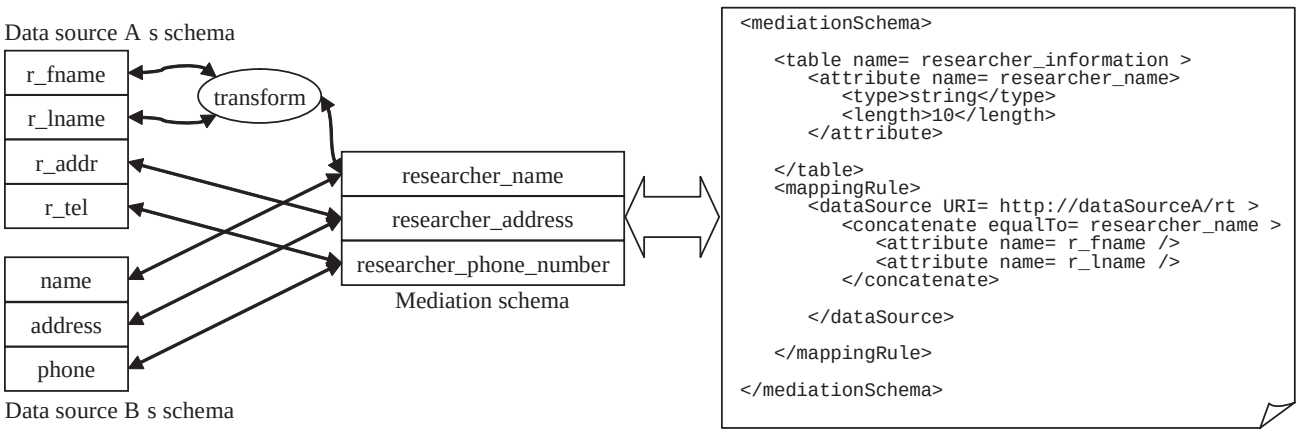
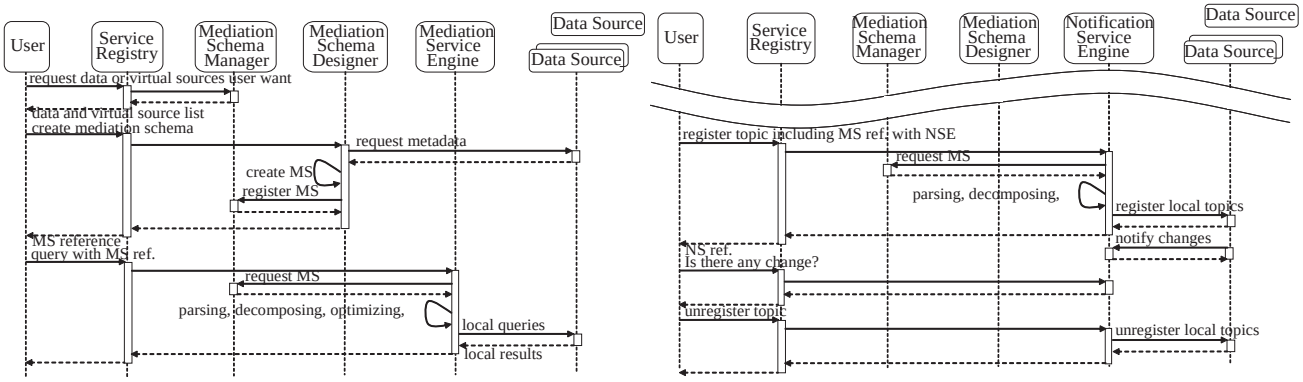


Figure 2. The example of a Mediation Schema



(a) Mediation Service **(b) Notification Service**

Figure 3. Conceptual sequence diagrams of Mediation and Notification services of DMSMS

3. The user registers a *topic* with *Notification Service Engine* (NSE). (The *topic* consists of MSR and some conditions that are used to check if data sources change.) NSE fetches mediation schema from *Mediation Schema Repository*. NSE parses user's *topic* and decomposes it into *local topics*. NSE registers these *local topics* with data sources and returns *Notification Service Reference* to the user.

4. When a change occurs in a data source, it is notified to NSE. NSE transforms the changed status into user's favorite form.

5. The user asks NSE whether a data source he has interest in have changed using *Notification Service Reference*. If NSE has been notified some changes, it returns them to the user.

6. The user unregister *topic* from NSE when he does not need topics any more. NSE also unregister *local topics* from data sources.

4 Technical Issues

In the previous section, we have analyzed users' requirements and conceptually designed DMSMS. In this section, we point out several technical issues that should be considered when developing an integration system based on the proposed architecture to increase system performance and/or offer users' convenience.

4.1 A Data Source Description Language and Its Query Language

DMSMS must help users to easily find data sources they need on the fly. In order to achieve this purpose, we should consider (i) *Data Source Description Language* (DSDL) that is used to describe data sources in a formal way, (ii) *Data Source Registry* (DSR) that stores and manages descriptions of data sources, and (iii) *a query language* and *query processor* for finding descriptions of needed data sources.

DSDL, defined by an XML format, allows owners of data sources to describe their data sources in a standard way. In order to find all data sources required by users, it should have expressive power enough to describe whole information of data sources such as databases, tables, attributes, and access methods. In addition, it must be able to represent information about replicas if data sources are replicated to improve performance and/or availability.

DSR stores and manages descriptions of data sources. Users check whether data sources exist by communicating with it. The physical storage of DSR can be based on RDBMS, XML DBMS, or other storage systems. The most important thing is that the storage system should be carefully designed to be efficient because it will store a large number of descriptions and process lots of queries sent by users.

DSR must support a query language for finding descriptions on data sources. If users write a query with the query language and send it to DSR, DSR processes it and returns the results. Because SQL is widely used, SQL-like query language will be desirable although query processing logic will be very different. Queries and results will be transmitted via Web Services.

In addition, virtual data sources presented by mediation schemas are also described by DSDL and the results are stored in *Internal Mediation Schema Manager*.

4.2 A Mediation Schema Definition Language

Mediation Schema Definition Language (MSDL) is an XML instance for defining mediation schemas. Users usually want selected data sources to be shown as one unified view. Using MSDL, users can create mediation

schemas that specify which attributes of which data sources should be included and what they should be named in the integrated view.

Even with a well-designed MSDL, it is difficult to make good mediation schemas because users have to specify much information into the schemas. In order to solve this problem, DMSMS should provide a GUI-based tool for users to easily define mediation schemas. This tool would automatically generate XML-based mediation schemas after the user visually designs an integrated data source with just a few drag-and-drops and mouse clicks.

Mediation Schema Repository (MSR) stores mediation schemas defined by MSDL and users find them in MSR using *Internal Mediation Schema Manager* (IMSM). IMSM should be designed in consideration of efficient storage systems, query languages, and query processors because the descriptions of virtual data sources are stored in IMSM.

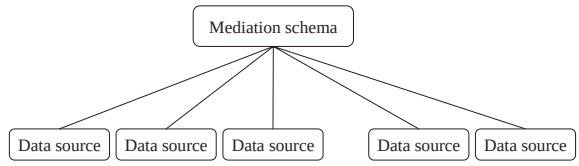
4.3 Use of Caches

By caching the query results, the system performance could be improved if there is a containment relationship between two queries. We define a query q contains another query q' if and only if the result set of q is the super-set of that of q'. In this case, if we cache the results of q, those of q' can be retrieved from the cache without re-accessing to the data source. Moreover, we can utilize the cached results.

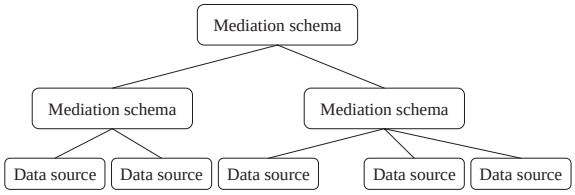
In order to improve the system performance by using cache, we should design mechanisms to determine if we can use the cached results instead of re-accessing to the data source. First, the mechanism checks the relationship between two mediation schemas referenced by two queries. Then, it checks if a containment relationship exists between two queries and if cached results can be used instead. The mechanism also has to solve traditional cache problems such as cache replacement and invalidation.

4.4 Replica Selection

A replica selection influences performance and availability of systems. As described in Section 4.1, if a data source has several replicas, DMSMS can recognize the fact that all of them represent the same data. Therefore, when DMSMS receives a query about the data source that has many replicas, it should be able to determine which replica is the best one to process the query. DMSMS will send the query to the selected replica in consideration of the network status, a load of each replica, and the cache status.



(a) Flat Mediation Schema



(b) Hierarchical Mediation Schema

Figure 4. Flat vs. hierarchical mediation schema

4.5 Automatic Mediation Schema Modification

Automatic mediation schema modification is needed to improve system performance and reduce maintenance costs. First, to improve system performance, we should choose flat mediation schema or hierarchical one presented in Figure 4 in consideration of all conditions such as query result caches, the number of data sources, and so on. If the hierarchical method is exploited, there are some advantages that users can easily define mediation schemas by using pre-defined schemas and the system can use caches that may be constructed on schemas in the lower level. Therefore, the system will process the query by automatically changing hierarchically defined schema to flat schema or vise versa according to the query execution plans. Second, in order to reduce maintenance costs, mediation schemas should be automatically modified when the schemas of data sources are modified. Maintaining mediation schemas is really labor-intensive because the schemas of data sources are often modified without notification. Therefore, DMSMS must periodically find changed schemas and automatically modify mediation schemas that consist of them.

4.6 Intelligent Data Integration Using Ontology

It is possible to construct *Intelligent Information Integration Systems* (IIIS) if data sources and mediation schemas are described by ontology languages such as Resource Description Framework (RDF) [21], RDF Schema (RDFS) [22], and Web Ontology Language [23]. In order to resolve heterogeneity, traditional systems transform some terms to other terms using mapping rules predefined by system administrators. These systems would deliver wrong results or could not operate at all unless mapping rules are defined. Therefore, these IIIS can integrate infor-

mation automatically without predefined mapping rules by exploiting ontologies.

5 Conclusions

We presented the design of a Grid-enabled Information Integration System based on mediator/wrapper, called *Dynamic Mediation Service Management System* (DMSMS). DMSMS searches data sources needed by users at run-time through *Data Source Registry*, and provides a GUI-based *Mediation Schema Designer* tool to help users to create mediation schemas easily and conveniently. Our system also provides a mediation service over virtual data sources (mediation schemas) and a notification service to actively deal with the changes of data sources.

We mentioned important technical issues that would occur when implementing a real product. They include (i) definition of *Data Source Description Language* (DSDL) and its *query language*, (ii) definition of *Mediation Schema Definition Language* (MSDL), (iii) use of caches for improving performance, (iv) replica selection for improving availability as well as performance, (v) automatic mediation schema modification, and (vi) intelligent integration of information using ontologies. We could develop more reliable and efficient system considering these issues.

Acknowledgements

This work was partially supported by *Korea Institute of Science and Technology Information* (KISTI) and *the Ministry of Information and Communication*, Korea, under the College *Information Technology Research Center* (ITRC) Support Program.

References

[1] A. Birrell and B.J. Nelson, "Implementing Remote Procedure Calls," *IEEE Computer*, Vol 25, No. 3, pp. 38-49, Mar. 1992.

[2] Java RMI, http://java.sun.com/products/jdk/rmi/

[3] DCOM, http://www.microsoft.com/com/tech/DCOM.asp

[4] CORBA, http://www.corba.org/

[5] I. Foster, C. Kesselman, J.M. Nick, and S. Tuecke, "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration," The Globus Alliance, Jun. 2002.

[6] I. Foster and C. Kesselman, *The Grid: Blueprint for a New Computing Infrastructure*, Morgan Kaufmann, 1998.

[7] GGF Database Access and Integration Service Working Group, http://www.cs.man.ac.uk/grid-db/

[8] M. Atkinson, R. Baxter, and N.C, Hong, "Grid data access and integration in OGSA," Data Access and Integration Working Group, Apr. 2002.

[9] OGSA-DAI, http://www.ogsadai.org.uk/

[10] G. Wiederhold, "Mediators in the architecture of future information systems," *ACM Transactions on Computer Systems*, Vol 2, No. 1, pp. 39-59, Feb. 1984.

[11] S. Busse, R.D. Kutsche, U. Leser, and H. Weber, "Federated information systems: concepts, terminology and architectures," Technical University of Berlin, 1999.

[12] A. Wöhrer and P. Brezany, "Mediators in the Architecture of Gird Information Systems," Institute for Software Science, Feb. 2004.

[13] A. Wöehrer, P. Brezany, and I. Janciak, "Virtualization of Heterogeneous Data Sources for Grid Information Systems," Proc. MIPRO, 2004.

[14] OGSA Distributed Query Processing, http://www.ogsadai.org.uk/dqp/

[15] M.N. Alpdemir, A. Mukherjee, N.W. Paton, P. Watson, A.A.A. Fernandes, A. Gounaris, and J. Smith, "Service-based distributed querying on the grid," Proc. ICSOC, 2003.

[16] J. Smith, A. Gounaris, P. Watson, N.W. Paton, A.A.A. Fernandes, and R. Sakellariou, "Distributed Query Processing on the Grid," *High Performance Computing Applications*, Vol 17, No. 4, pp. 353-367, 2003.

[17] National Technical Information System, http://www.kisti.re.kr/english/03_department /info_depart_01.jsp?mid=03_01

[18] Ministry of Science and Technology, http://www.most.go.kr/en/sce02/sce0201/

[19] University of Vienna Institute for Software Science, The Knowledge Grid Project, http://www.gridminer.org/

[20] myGrid project, http://www.mygrid.org.uk/

[21] Resource Description Framework (RDF), http://www.w3.org/RDF/

[22] RDF Schema, http://www.w3.org/TR/rdf-schema/

[23] OWL Web Ontology Language, http://www.w3.org/TR/owl-features/