Heterogeneous Distributed Databases and Distributed Query Processing in Grid Computing

Ǧorgi Kakaševski[1], Anastas Mišev[2] and Boro Jakimovski[2]

[1] Faculty of Information-Communication Technologies, FON University,
bul. Vojvodina b.b. 1000 Skopje, R. Macedonia
{gorgik}@fon.edu.mk
[2] Institute of Informatics, Faculty of Sciences and Mathematics,
Ss. Cyril and Methodius University, 1000 Skopje, Republic of Macedonia
{anastas, boroj}@ii.edu.mk

Abstract. Access to large data volumes is one of the stronger benefits of the Grid computing. While most of this data is file based, the need for access to structured data is increasing. Because of the nature of Grid to be heterogeneous and distributed environment, the database systems which should works on Grid must support this architecture. These systems must also have a solution for distributed query processing, for retrieving data from different sites and in different formats. OGSA-DAI is an example of such extensible service-based framework that allows data resources to be incorporated into Grid fabrics. As a conclusion of this paper, we propose several visions for future work.

Keywords: Grid computing, heterogeneous databases, distributed query processing, OGSA-DAI.

1 Introduction

The computational needs of today's science and engineering are constantly increasing. The Grid provides both computational and storage resources to the scientific applications [1].

E-Science is computationally intensive science that is carried out in highly distributed network environments, or science that uses immense data sets [2]. It incorporates all of the traditional sciences: earth sciences, physical sciences and life sciences, but also includes medical research and diagnosis, engineering etc. Digital data play major role in e-Science applications, from small projects [3] to The Large Hadron Collider [4]. This data are organized as shared and structured collections, which are stored in databases, in structured documents or in assemblies of binary files. Because this requirements of Grid applications, developers of Grid systems, should provide practical implementations of Grid databases (or data access integration [5]). Another reason to implement databases on Grid is the Grid itself. It uses many organized data for its own functioning.

The rest of the paper is organized as follow. In Section 2 we describe the key questions of implementing heterogeneous database on Grid, and in Section 3 we

describe query processing in distributed environments. In Section 4 we present practical implementation of database middleware for Grid, OGSA-DAI, and in Section 5 we mention another projects which cover this topic. We give vision for future work in Section 6 and we conclude paper in Section 7.

2 Heterogeneous Distributed Databases

In this Section, we want to highlight several questions that are important for implementation of heterogeneous distributed database system on Grid.

One of the main characteristics of the Grid is it heterogeneity [6], which means that Grid is made of different type of resources with different characteristics, and as whole must act as one system. For example, on the Grid we can find two clusters with different type of processors or installed software, even on the same cluster we can find machines with different resources. This property is true for databases. There are several types of databases which can be used in applications: relational, XML, even CSV files or just files [7]. But The Grid users (applications, clients) must see data as one integrated part, no matter where they are located or which is their physical form.

Another key moment in organizing databases on Grid is the service-oriented architecture of Grid and Grid services, defined by Open Grid Service Architecture (OGSA) [8]. OGSA has a one of the central roles in Grid computing, because it simplifies the development of secure, robust systems and enable the creation of interoperable, portable, and reusable components. The main idea of transient Grid services is to extend well-defined Web service standards (WSDL, XML schemas, SOAP, etc.), to support basic Grid behaviors.

On Fig. 1 is given the proposed messages flow between client and data stored on Grid, by Global Grid Forum in Grid Database Service Specification [5]. This scheme enables clients to access or modify the data and their organization.

Grid Data Service is one kind of transparent wrapper. Clients exchange data with database management systems through Grid Data Service, which acts as "proxy". In addition, Grid Data Services must support common database standards, such as SQL query language [9], and to be in line with Grid system, to use Grid authentication and authorization mechanisms [10] etc. Grid Data Service Factory is used for creating Grid services (instead of Web services, where factory is not needed).

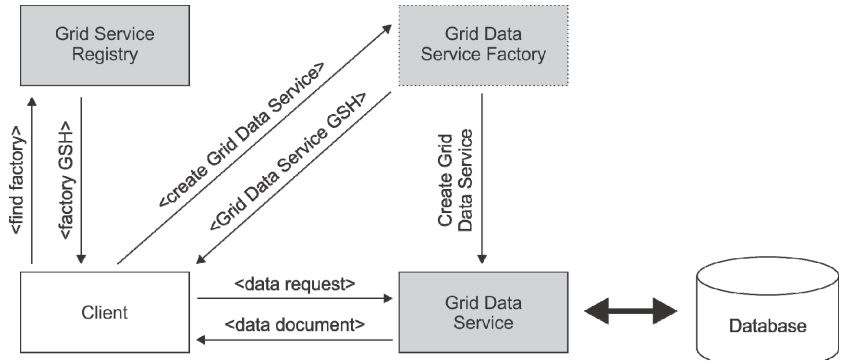

Fig. 1. Accessing and modifying data stored on Grid using a Grid Data Service

Because data can be located on many sites and in different physical forms, on Fig. 2 is presented one of the possible Grid database architectures. According to this, Grid nodes are installed with database management systems (DBMS).

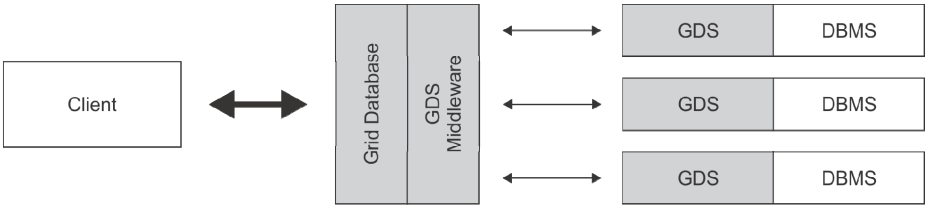

Fig. 2. Architecture of Grid Database System

During the fact that these traditional DBMSs exist for many years and are well developed and tested, in our case we can use some of the existing. For example, DBMS can be MySQL, Oracle or XML. On top of this system Grid Data Services must be implemented, as part of the middleware that is responsible for communications among heterogeneous distributed database.

Grid Data Service Middleware is one kind of transparent wrapper for complex Grid database [11]. Transparency refers to: location, name of database, distribution, replication, ownership and cost, heterogeneity and schema [12]. Clients exchange data with Grid database, even if they don't know where that data is stored or in which format. Important for client is that in Grid database must be implemented all (or common) functionalities of regular database management systems and in natural way. That ensures that clients (users, software developers) can easily use Grid database and easily adapt their applications with new kind of database.

As a conclusion, a heterogeneous distributed database system (or simply Grid database) should provide service-based access to existing database management systems that are located on Grid nodes. This database will be some form of virtual

database and clients will access to data on transparent way no mater of physical organization of data [13].

However, there are many other ways to implement databases on Grid, for example, to keep all data in main memory, to use centralized client-server architecture [14], or to use Remote Procedure Call based service. But these types of organizing data on Grid will face with many problems when applications work with huge amount of data, and also can't take the advantages of The Grid.

3 Distributed Query Processing

Distributed query processing (DQP) is natural topic when we discuss about databases on Grid [15]. Because the fact that databases are distributed, and data can be found on many locations, the need of effective and efficient DQP is very important. For example, how to do join in Grid environment where two tables with enormous data are on different locations? One approach is to try to implement standard DQP techniques in Grid environment. Of Fig. 3 is shown standard steps in query processing which are described in [16].

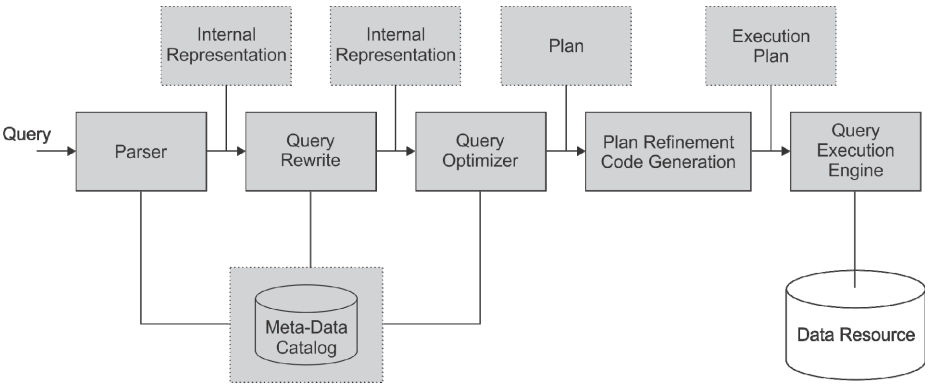

Fig. 3. Query processing steps

DQP should extend or improve these elements. Except the need of Grid database (schema integration, multiple vendor support, administration facilities etc.), DQP for Grid should satisfy this requirements:

- Table partitioning: A table can be dynamically decomposed into smaller parts, horizontally or vertically, and stored on the Grid at many locations.
- Multiple data format support: Non-relational resources should also be included in SQL joins.
- Multiple Site Support: DQP allows the data to be delivered to several different locations on different ways (FTP, GridFTP, even via e-mail).

Supporting all the functionalities above is not a trivial task. A lot of questions are open, such as parallel search, sort, group by, join, scheduling, optimization and DQP researchers always looking for improvements. More can be found in these studies [17][18][19].

4 Practical implementation with OGSA-DAI

OGSA-DAI stands for Open Grid Services Architecture - Data Access and Integration [20]. It is a Java open source project and it is a one of the most comprehensive heterogeneous database system, which can be used in Grid.

One of the primary goals of data access and integration is to treat these data, whatever their origin, within a uniform framework. OGSA-DAI makes the DBMSs, which can be heterogeneous and distributed on Grid, transparent to the users [21]. It hides the underlying drivers that are needed to communicate with the database management and as name tells, OGSA-DAI provide service-oriented interface to the clients. This process naturally will add some overhead on top of the communication itself but it is not that significant [22].

It supports several kinds of databases: relational (Oracle, DB2, SQL server, MySQL, Postgress), XML databases like xindice and eXist and files. There is an opportunity for programmers to develop new drivers for DBMS which are not supported in original version. Acting in the top of local DMBS that are located on different sites, OGSA-DAI provides uniform Grid database.

Today, OGSA-DAI has a several thousand users and a lot of scientific projects that are using it as a Grid database (i.e. ADMIRE1, BIRN2, GEO Grid [23], MESSAGE [24], BEinGRID5, LaQuAT6, Database Grid7, mantisGRID [25] etc.). List of most important projects can be found in [26].

4.1 Architecture

Architecture of OGSA-DAI is in compliance with needs described in Section 2. In OGSA-DAI, data resource represent externally managed database system and data service provide interface to access to this data resources.

Activities [27] are a units of work and they perform well defined data-related task over a specific resource. Main activities are organized in three types: Statement Activity, Translation Activity and Delivery Activity (Table 1). Group of activities form query-transform-deliver workflow, and users interact with OGSA-DAI services by submitting workflows to Data Request Execution Service (DRES) [28]. On Fig. 4 a) is shown example of OGSA-DAI workflow with three activities.

Table 1. List of main activities in OGSA-DAI

Type of activity	Example
Relational Database	SQLQuery, SQLUpdate, SQLBulkLoadTuple, SQLParameterisedQuery, ExtractTableSchema,
XML Database	XPathQuery, XQuery, XUpdate, AddDocument, GetDocuments, RemoveDocuments
Transform	ByteArraysToTuple, CSVToTuple, XSLTransform, StringTokenize
Delivery	DeliverToDataSink, DeliverToFTP, DeliverToSMTP, DeliverToGFTPExtended, DeliverToSession, ObtainFromFTP
DQP	CreateDQPResource, ExtractTableSchema (DQP), SQLQuery (DQP)

It is important to understand the interaction between client and data through DRES, shown on Fig. 4 b).

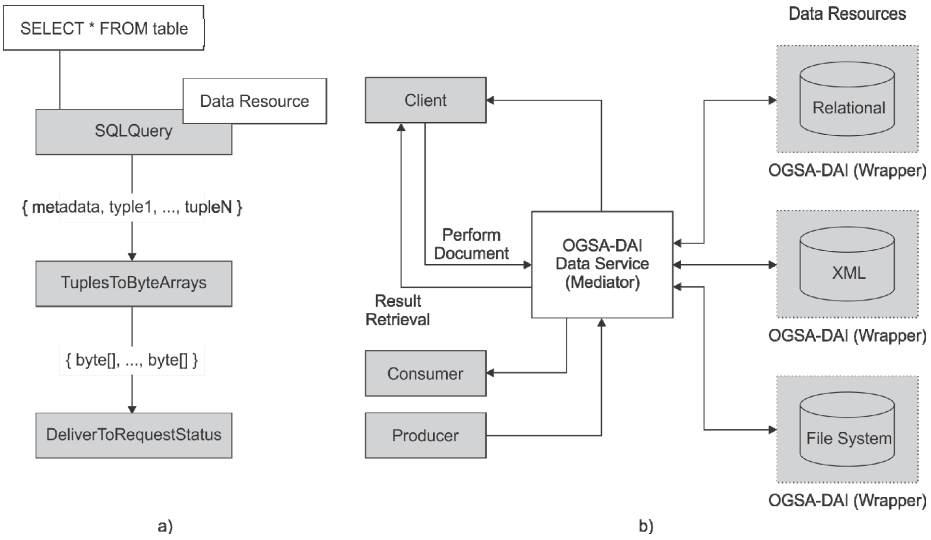

Fig. 4. a) Example of OGSA-DAI query-transform-deliver workflow. b) Interaction between client and data.

The first step in this process is sending perform document from client to DRES. This document is XML based document which include sequence of activities. This activities are well defined tasks which will be completed on server side and this activities are connected one another to form a workflow where output from one activity is an input to another. Finally, result data can be delivered back to the client, or to third party (consumer).

4.2 OGSA-DQP

The service-based OGSA-DQP [29] approach functions as an integrated component in OGSA-DAI. It enables distributed queries over relational data resources. Although the core data service in OGSA-DAI provide abstraction for accessing individual data resources on the Grid, they do not address challenges associated with integrating data from multiple resources.

DQP coordinator service and DQP evaluator service are two services that extends OGSA-DAI. The first one have a DQPQueryStatement activity which compiles, optimizes and schedules SQL queries for execution. The DQP evaluator service is capable of evaluating a query fragment provided by the coordinator. Evaluator communicate with OGSA-DAI resources, and to provide benefits of parallelism, multiple evaluators are used.

On the beginning, OGSA-DQP obtains the metadata that is needs to compile, optimize, partition and schedule distributed query plans over the multiple execution nodes (evaluators). After the submitting of query, the process is divided in two steps:

(i) the query plan is produced and partitioned, and each partition is sent to the relevant evaluator; (ii) the evaluators retrieve data from OGSA-DAI data services and invoke any analysis services required to evaluate query. Evaluator is exploiting partitioned parallelism both for query-internal operations (such as join) and for external web service calls.

In the newest version (OGSA-DAI 4.1), published in the time of writing, there are some changes and improvements [28]. Major change is that DQP evaluator service is removed, an all of its functionality is now provided by new activities in OGSA-DAI. New version support more SQL operations, improve extensibility, introduced new way of configuring. However there are still shortcomings and some of them are mentioned in Section 6.

5 Related Projects

More early works on organizing data on Grid focused on file-based data and metadata describing files. Several projects made effort to support structured data collections.

Spitfire [30] is a part of European Data Grid project. It use GSI-enabled HTTP for delivering data to clients and has developed an infrastructure that allows a client to query a relational database. An XML-based protocol is used to represent the query and its result. The key feature of the system is its support for role-based security, which builds upon Grid certificates: clients can specify the role they wish to adopt for a query execution, and a mapping table in the server checks that they are authorized to take on this role.

The Storage Request Broker (SRB) [7] is a middleware that provides uniform access to datasets on a wide range of different types of storage devices that can include file systems and archival resources. The SRB's primary focus is on files and collections of files (definition of dataset is 'stream-of-bytes' and can be held as Large Objects - LOBs). As middleware it provide location transparency, because it use metadata that allows access to datasets by logical names and other metadata attributes. Replica management for dataset is one of the priorities of SRB and if the primary storage is unavailable, SRB provide fault tolerance by redirecting client request to a replica. SQL query can be registered as an object with the SBR - the query is executed whenever the object is retrieved.

There are also several projects developed in the field of DQP on Grid. Polar* [31] is accepted to be the first study to employ distributed query processing in a Grid environment, but it is not service-based. It exploits Polar system, which is a parallel object database server, with facilities of DQP by using MPICH-G [MPI] (i.e. grid enabled MPI). Polar* consists of the parser, logical optimizer, physical optimizer, partitioner, scheduler and evaluator modules. Generation and evaluation of efficient query execution plans for OQL [32] is also responsibility of Polar*. This projects are also in pre-service-based grids: GridDB [33], GridDB-Lite [34] and POQSEC [35].

There are several other service-based DQP systems: SkyQuery [36], Web Service Management System (WSMS) [37], Garlic [38] and Kleisli [39]. SkyQuery works mainly with astronomical federated data but it is still a good application of service-oriented DQP. It consist of three components: Clients, Portal and SkyNodes. Upon

receiving the query from Client, Portal generates performance queries by decomposing the query and then using the results of these performance queries, creates an optimized execution plan to be sent to SkyNodes for evaluation. WSMS allows clients to query multiple web services simultaneously using an SQL-like interface in a seamless and integrated manner while benefiting from pipelined parallelism. This approach use distributed Web services, but all other operations (such as joins) take place at centralized WSMS.

6 Vision for Future Work

Although today we have several implementations of Grid databases, this area is relatively new and there is a need for future work.

Database partitioning is one of the questions that should be solved in transparent way, moreover to make this process dynamic by Grid. Partitioning should split tables on different locations (horizontally or vertically), but gives a non-partitioned database view for the client side when working with SQL commands. There are several benefits from database partitioning: manageability, scalability, performance, availability and security.

There is a need for developing a new tools for monitoring, replica management, fault-tolerance, QoS [40] etc.

The projects mentioned before, as a good example of Grid database system, OGSA-DAI, can also be extended. The structure of this open source project is made to be extensible in many different ways by: writing activities, writing data resources, writing application-specific presentation layers, to improve or add functionalities to security or database access. This enables third parties to continue the development of stabile database system with functionalities, instead of developing a new one.

OGSA-DQP also has some limitations. Parallel query-processing optimizations utilizing additional evaluators are not yet supported. There is no GUI client (web based user interface). All the existing optimizers are essentially heuristic and do not make use of a cost model, join ordering makes use of very basic cardinality estimates, there is no dynamic fault tolerance, etc.

Existing Grid applications can be also updated to support some of existing Grid database systems. Except of traditional database techniques, which can be used in data intensive applications, for organizing, holding and querying data, there are many other topics of interest that can be implemented in Grid databases. Some of them are parallel online analytic processing [41], parallel data mining [42], parallel clustering and classification etc.

7 Conclusion

We can highlight three conclusions. First, Grid applications have an increasing need of database systems. Combining Grid and database technologies is an essential approach to meet the requirements of large-scale Gird applications. Almost every application running on Grid has many requirements for access to structured data, and

Grid as a platform with their resources can provide many benefits for such king of database system.

Our second conclusion is that today there are many efforts to enable organizing and querying data in Grid environment. OGSA-DAI is a good example. And, finally, there is a still need for improvements and space for discovering new technologies for working with large amounts of data.

References

1. I. Foster, C. Kesselman. The Grid: Blueprint for a New Computing Infrastructure. Elsevier, 2004.
2. J. Taylor, Defining e-Science, http://www.nesc.ac.uk/nesc/define.html.
3. G. Kakasevski, et al., Grid enabled system for gathering and retrieving images from WEB, ETAI 2007.
4. The Large Hadron Collider Homepage. http://lhc.web.cern.ch/lhc/.
5. N.P.C. Hong, et al., Grid Database Service Specification, GGF Informational Document, Feb. 2003.
6. Foster, I., Kesselman, D. and Tuecke, S. The anatomy of the grid: enabling scalable virtual organizations. International Journal of Supercomputer Applications, 15(3), 200-222. 2001.
7. Rajasekar, A. Wan, M. and Moore, R. MySRB & SRB - Components of a data grid. 11th International Symposium on High Performance Distributed Computing (HPDC-11), Edinburgh, 2002.
8. I. Foster, C. Kesselman, J. M. Nick, and S. Tuecke, "The physiology of the grid: An open grid services architecture for distributed systems integration," in Open Grid Service Infrastructure WG, Global Grid Forum, vol. 22, pp. 1–5, Edinburgh, 2002.
9. ISO/IEC 9075-1:2008: Information technology – Database languages – SQL – Part 1: Framework (SQL/Framework).
10. http://www.ietf.org/rfc/rfc2459.txt, Internet X.509 Public Key Infrastructure Certificate and CRL Profile (Request for Comments: 2459), Network Working Group, Janyary 1999.
11. Shiers, J. Building a multi-petabyte database: The RD45 project at CERN, in Loomis, M.E.S. and Chaudhri, A. B. Object Databace in Practice. Prentice Hall, pp. 164-176.
12. Atkinson, M.P., Dialani, V., Guy, L., Narang, I., Paton, N.W., Pearson, D., Storey, T. and Watson, P. Grid Database Access and Integration: Requirements and Functionalities, DAIS-WG, Global Grid Forum Informational Document (GFD.13), March 2003.
13. A. Lee, J. Magowan, P. Dantressangle and F. Bannwart. Bridging the Integration Gap, Part 1: Federating Grid Data. IBM Developer Works. August 2005.
14. Kakashevski, G., Loskovska, S., Buckovska, A., Dimitrovski, I.; Grid Enabled Medical Image Gathering System; ITI 2007. 29th International Conference on 25-28 June 2007.
15. A. Gounaris. Resource Aware Query Processing on the Grid. Ph.D Thesis. 2005.
16. D. Kossmann, "The state of the art in distributed query processing," ACM Computing Surveys (CSUR), vol. 32, no. 4, pp. 422–469, 2000.
17. C. T. Yu and W. Meng, Principles of Database Query Processing for Advanced Applications, The Morgan Kaufmann Series in Data Management Systems. 1997.
18. J. Hellerstein et al, Adaptive Query Processing: Technology in Evolution, IEEE Database Engineering Bulletin, Vol. 23, No. 2, pp. 7-18. 2000.
19. A. Deshpande, et al. Adaptive Query Processing: Why, How, When, What Next. In Proc. ACM SIGMOD 2006, pp. 806-807. 2006.
20. M. Antonioletti, M. Atkinson, R. Baxter, A. Borley, N. Hong, B. Collins, N. Hardman, A. Hume, A. Knox, M. Jackson, et al., "The design and implementation of Grid database

services in OGSA-DAI," Concurrency and Computation: Practice and Experience, vol. 17, no. 2-4, pp. 357–376, 2005.
21. Jackson, M. et al. OGSA-DAI 3.0–the whats and the whys. In Proc. UK e-Science All Hands Q6 Meeting pp. 158–165, 2007.
22. Kumar Abhinav. Evaluation of Grid Middleware OGSA-DAI. School of Computing Science, Vellore Institute of Technology, Deemed University. May 2006.
23. Tanimura Y, Yamamoto N, Tanaka Y, Iwao K, Kojima I, Nakamura R, Tsuchida S and Sekiguchi S. Evaluation of Large-Scale Storage Systems for Satellite Data in GEO GRID. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. Vol. XXXVII. Part B4. pp 1567-1574. Beijing 2008.
24. MESSAGE (Mobile Environmental Sensing System Across Grid Environments) Project. http://bioinf.ncl.ac.uk/message/. 2010.
25. Garcia Ruiz, M., Garcia Chaves, A., Ruiz Ibañez, C., Gutierrez Mazo, J.M., Ramirez Giraldo, J.C., Pelaez Echavarria, A., Valencia Diaz, E., Pelaez Restrepo, G., Montoya Munera, E.N., Garcia Loaiza, B. and Gomez Gonzalez, S. mantisGRID: A Grid Platform for DICOM Medical Images Management in Colombia and Latin America. J Digit Imaging. Feb 2010.
26. OGSA-DAI open source project publications. http://sourceforge.net/apps/trac/ogsa-dai/wiki/Publications. 2011.
27. List of OGSA-DAI Activities, http://ogsa-dai.sourceforge.net/documentation/ogsadai4.1/ogsadai4.1-axis/ActivitiesReference.html. 2011.
28. Dobrzelecki B, Krause A, Hume A, Grant A. Antonioletti M, Alemu Y, Atkinson M, Jackson M, Theocharopoulos E (2010) Integrating distributed data sources with OGSA–DAI DQP and Views. Phil. Trans. R. Soc. A 13 September 2010 vol. 368 no. 1926 4133-4145. 2010.
29. Lynden, S., Mukherjee, A., Hume, A. C., Fernandes, A. A., Paton, N. W., Sakellariou, R., and Watson, P. 2009. The design and implementation of OGSA-DQP: A service-based distributed query processor. Future Gener. Comput. Syst. 25, 3 (Mar. 2009), 224-236. 2008.
30. W. H. Bell, D. Bosio, W. Hoschek, P. Kunszt, G. McCance, and M. Silander. Project Spitfire - Towards Grid Web Service Databases. In Global Grid Forum 5, 2002.
31. J. Smith, A. Gounaris, P. Watson, N. W. Paton, A. A. A. Fernandes, and R. Sakellariou, "Distributed query processing on the grid," in GRID '02: Proceedings of the Third International Workshop on Grid Computing, (London, UK), pp. 279–290, Springer-Verlag, 2002.
32. Cattel, R. Object Database Standard: ODMG 2.0. San Francisco, CA: Morgan Kaufmann Publishers. 1997.
33. D. Liu and M. Franklin, "GridDB: A data-centric overlay for scientific grids". 2004.
34. S. Narayanan, T. Kurc, U. Catalyurek, and J. Saltz, "Database support for data-driven scientific applications in the grid," vol. 13, pp. 245–272, 2003.
35. R. Fomkin and T. Risch, "Framework for querying distributed objects managed by a grid infrastructure," in Data Management in Grids, pp. 58–70, Springer-Verlag, 2006.
36. T. Malik, A. S. Szalay, T. Budavari, and A. R. Thakar, "Skyquery: A web service approach to federate databases," in Proc. 1st Biennial Conference on Innovative Database Systems Research (CIDR), 2003.
37. U. Srivastava, K. Munagala, J. Widom, and R. Motwani, "Query optimization over web services," in VLDB '06: Proceedings of the 32nd international conference on Very large data bases, pp. 355–366, VLDB Endowment, 2006.
38. V. Josifovski, P. Schwarz, L. Haas, and E. Lin, "Garlic: a new flavor of federated query processing for db2," in SIGMOD '02: Proceedings of the 2002 ACM SIGMOD international conference on Management of data, (New York, NY, USA), pp. 524–532, ACM, 2002.

39. S. B. Davidson, J. Crabtree, B. Brunk, J. Schug, V. Tannen, C. Overton, and C. Stoeckert, "K2kleisli and gus: Experiments in integrated access to genomic data sources," IBM Systems Journal, vol. 40, pp. 512–531, 2001.
40. Anastas Misev, Emanouil I. Atanassov: User Level Grid Quality of Service. LSSC 2009: 507-514.
41. David Taniar, Clement H. C. Leung, Wenny Rahayu, Sushant Goel. High Performance Parallel Database Processing and Grid Databases. Wiley Series on Parallel and Distributed Computing.
42. V. Stankovski, et al., Grid-enabling data mining applications with DataMiningGrid: An architectural perspective, Future Generation Computer Systems. 2007.