Current Development in Artificial Intelligence.
ISSN 0976-5832 Volume 2, Number 1 (2011), pp. 51-63
© International Research Publication House
http://www.irphouse.com

Intelligent Knowledge based Heterogeneous Database using OGSA-Dai Architecture

[1]Appasami G. and [2]Karthikeyan S.

[1]Assistant Professor, Department of CSE, Dr. Pauls Engineering College,
(Affiliated to Anna University Chennai), Chennai, India
E-mail: appas_9g@yahoo.com
[2]Assistant Professor, Department of IT, Bannariamman Institute of technology
(Affiliated to Anna University Coimbatore) India
E-mail: karthikeyan_123s@yahoo.co.in

Abstract

In this paper we present a framework to manage the distributed and heterogeneous databases in grid environment Using Open Grid Services Architecture – Data Access and Integration (ODSA-DAI). Even though there is a lot of improvement in database technology, connecting heterogeneous databases within a single application challenging task. Maintaining the information for future purpose is very important in database technology. Whenever the information is needed, then it refers the database, process query and finally produces the result. Database maintains the billion of information. User maintains their information in different database. So whenever they need, they collect it from different database. User cannot easily collect their information from different database without having database knowledge. The current database interfaces are just collecting the information from many databases. The Intelligent Knowledge Based Heterogeneous Database using OGSA-DAI Architecture (IKBHDOA) provides solution to the problem of writing query and knowing technical details of Database. It has intelligence to retrieve the information from Different Sets of Database based on user's inputs.

Keywords: Intelligent Knowledge Base (IKB), Heterogeneous Database, Automatic Query Generation (AQG), Relationship between database tables.

Introduction

The database is very useful to maintain the huge information. Now a days many organizations requires database to maintain their information electronically. Whenever user needs to collect information or manipulates information, user has to write complex query in the database query language, so user needs to have more knowledge in database and its query language. The Intelligent Knowledge Based Heterogeneous Database using OGSA-DAI Architecture (IKBHDOA) provides a user interface through which user can easily access the multiple databases without writing query and without knowing internal details of database [1] [9].

OGSA-DAI

The Open Grid Services Architecture – Data Access and Integration (OGSA-DAI) aims to provide a middleware solution to provide access to and integration of data for applications working across several database domains. Early Grid applications focused principally on the storage, replication, and movement of file-based data, but many applications now require full integration of database technologies and other structured forms of data through Grid middleware. OGSA-DAI provide activities to access relational, XML databases, and indexed files. It also provides data translation and third-party delivery activities. This activity framework provides extensibility so that application developers can add their own activities. This paper describes the architectural requirements for future use, the new architecture's functionality and components and presented in detail [2] [3].

Architecture

The architecture is consists of several components, they are metadata manager, transaction manager, Query manager, notification manager and scheduling manager with data sources for various databases. The architecture is described in Figure 1.

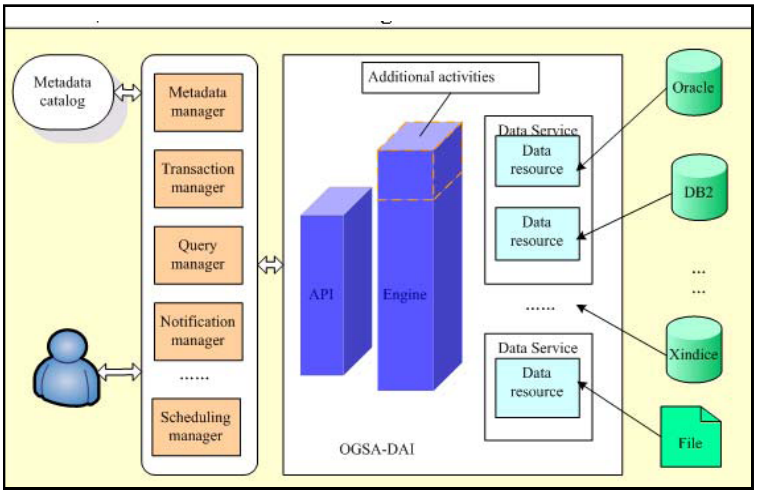


Figure 1: Architecture.

The detailed functions of the components are described in the following section

[5] [6].

Components

OGSA-DAI is a project that develops middleware to assist with access and integration of data from separate sources via the grid. OGSA-DAI includes a collection of components for querying, transforming and delivering data in different ways, and a simple toolkit for developing client application, it focuses on data accession, rather than data integration [4] [5]. We build our system on the top of OGSA-DAI, benefiting from the following reasons:

1. The hiding of heterogeneity.
2. The scalable framework, we can extend the functionalities by adding user-defined activities.
3. Providing a uniform interface, accessing data resources in grid by the form of service.
4. Including basic activities of statement (e.g. SQL query), transformation (e.g. compress) and transport (e.g. Grid FTP).

Adopting OGSA-DAI as the foundation sharply reduced the complexity.

Additional activities

The additional activities we defined are foundation units for managing grid database, they are organized in catalogs, which can be described in figure 2, the rectangles on the left side of dashed line represent the basic activity catalogs that OGSA-DAI provided (e.g., statement), the rectangles on the right side represent the additional activity catalogs we developed (e.g., transaction), each catalog includes a series of activities around the catalog name. Practically, most of the products of DBMS support functions of metadata manipulation, transaction control etc, to them [7] [10]. They are discussed below:

Metadata manager

The metadata manager is a core component of the system, it is responsible for collecting and managing the data resources metadata, constructing integrated data schema and conducting schema mapping. First, it could collect and manage the metadata of the data resource by the functionality offered by metadata activity. Second, two schemes are included to construct integrated data schema. One scheme is constructing global schema on the top of all the data resources, we call it global scheme, global scheme is for users that hoping to use our framework as information integrated system (giving some keywords like “protein”, querying for all useful information, then, the system returning the results in some specified order like relativity), which works as an extended search engine. The other scheme is part scheme, there is no global integrated schema but part integrated schema provided in this scheme, this is for users who would like to customize workflow (they know the structure and semantics of low-level data schemas of the data resources in the system, they define the part integrated schema for their specify motivation). The two different schemes satisfy different requirement. Global scheme need little workload for end users but enormous workload for administrator to construct global schema, importing automatic schema matching mechanism [8] is our next plan to reduce the overhead. Part scheme is much flexible, different users could design their own part integrated schema for their specified need, it is impossible to design a global schema that could perfectly fit to a particular user's information need and emphasize his individual domain of research. Third, schema mapping rules are recorded and managed by metadata manager, it would conduct transformation between integrated schema query and respective data resources queries. Metadata manager is the essence of constructing and managing data integration system, the assumption of data schema evolution is focus on this template [8] [9].

Query manager

Query for a single database could be accomplished by the basic activities of statement activities provided in OGSA-DAI. Distributed query processing need the coordinated work of metadata manager, query manager, transaction manager, OGSADAI data services and etc, query manager works like the coordinator during this process, the query processing is a two-step process, it firstly parses the query into a logical query plan by calling the metadata manager, and then execute the query plan. Different query algorithms could be develped in query manager, transaction manager is employed to ensure the ACID properties during the procedure.

Transaction manager

In transaction manager, local transaction of the data resource can be implemented using transaction activities, global transaction implemented by coordinating the execution of various transactions on data resources. To adapt the dynamic grid, transaction mechanism with different strictness and granularity should implement, the transaction manager are indispensable for ensuring correctness of sharing information and cooperating work.

Scheduling manager

Gird environment is dynamic, adaptively, dynamically scheduling the grid resources (network bandwidth, storages, and etc) could improve performance, availability and efficiency. To make scheduling decision, scheduling managers should make interaction with grid monitor service to get static and dynamic information about computer nodes (e.g., CPU, memory, disk) and network (e.g., bandwidth and latency).

Notification manager

This would allow users to register some interest in changes of a set of data, updates of integrated schema, switches of request state, it includes mechanisms for users to specify what it is interested in and a method for notifying the users for notifying the users the change. Metadata catalog is a basic component in grid, It keeps the metadata and provides a mechanism for storing and accessing metadata. The metadata information related to this framework includes:

1. Metadata of OGSA-DAI services and data resources.
2. Integrated schema of the data resources in the system.

Data resource Consists of a set of structure (e.g. Oracle, DB2), semi-structure and unstructured data resources [8] [17].

Intelligent Knowledge Base

The Intelligent Knowledge Base (IKB) provides an interface between the user and database without writing query. Any user can easily get the information from heterogeneous multiple data base. It provides sophisticated user interface so that a naive user can easily work on several database.

User must write different query for different database. This brings lot of complexity while accessing the information from different database. They must write the separate query for each database and collects the result. The IKB easily collects the information from different database and shows in the result window [2] [5].

Proposed System

Our proposed system is developed using OGSA-DAI and The Intelligent Knowledge Base. The Intelligent Knowledge Base for Heterogeneous Database using OGSA-DAI Architechture using OGSA-DAI Architechture (IKBHDOA) provides an interface between the user and database. Without writing query user can easily get the information from single or multiple data base. Our new system provides User interface in such a way that any user with out database knowledge can also work on multiple hetereogenous data bases.

Problem Statement

The various domains like Medical, Agriculture, Mechanical industries etc. are maintaining their information in single or multiple databases for their own usage. To retrieve information from database, complex queries and some complex procedures must be used, so these organizations need the technical person to do all kind of data base management activities. This will add extra operational cost to their expenses.

Gathering the information from single database requires the basic and special knowledge in database and its query languages. In case if information is scattered across the multiple databases, then it needs some complex method to retrieve the information from different databases. These are the problems to that organization, whenever they maintain their information in Data base.

This system needs the following input from the user. Those inputs are Database name, Datasource name, User name and Password. By using this information, The IKBHDOA connects to the single or multiple databases and collects table Names and their column Names and additional information like primary key, Foreign key, column’s datatype etc. All these information can be viewed as tree structure with an icon format. The user just drags the table or column name from the Tree view in the display panel and drops them in to the execution panel. The filter conditions can also be set for selected columns [14] [15].

The intelligent core in background collects the relationships among the selected table items. After the selection, the query is generated and then executed, and the final result is displayed in output form. The IKBHDOA also provide option to the user to temporarily maintain the result in the table format, so user can uses this result as a table for some other operations.

Design

The IKBHDOA user interface has three panels. Those panel names are Display panel, Execution Panel and Result Panel. The Display panel shows the tables and their columns information along with column properties. The Execution panel has user selected table items. The result panel has the final result. The overall IKBHDOA architecture is shown in figure 2.

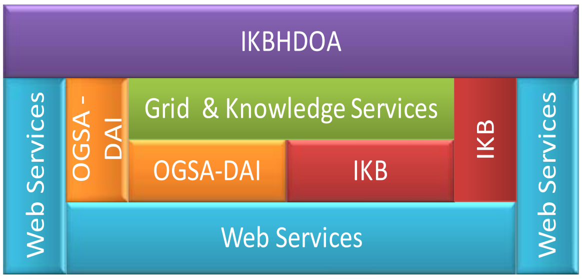


Figure 2: IKBHDOA Architecture.

The IKBHDOA collects the user's database information like Database name, Datasource name, Username and Password from the user. By using that information it creates a connection with the user's database. After connection is created, it collects the tables and their columns information from the database and maintains that information in the local system. It displays that (tables and their columns) information in display panel.

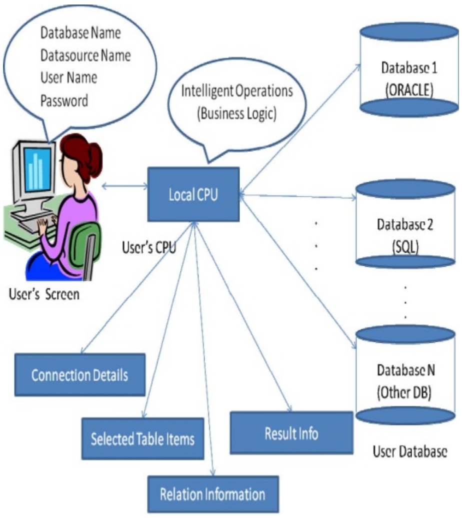


Figure 3: Pictorial representation of the IKBHDOA System.

User can just drag the table or column name from display panel and drop them into the execution panel. User can also set the additional condition on the selected column. These user selected items and user conditions (Filter condition) are kept in the local system. The above Figure 3 shows Pictorial representation of the IKBHDOA System.

The intelligent component runs in background. It first identifies the relationship among the selected items and forms the query. After the query is generated, query is sent to database through the database connection. Then database system executes that query. The Result is sent back to the IKBHDOA.

The IKBHDOA gets the result from the database and stores the information in the local system. Then the result is displayed in result panel [10] [11] [12].

Module Design

The IKBHDOA is implemented in module basis. It has three main modules. Those modules are Interface Module, Business Logic Module and Database Module. Each module does different functionality and statically coupled together.

Interface Module

The interface module performs all user interface related operation. This module creates and maintains the display panel, execution and result panel. It has the database connection form that form is used to get the user's database information from end user.

It passes database information to the Business Logic Module. Then it gets tables and their column information from the Business logic module. This information is displayed in the display panel by this module.

This module provides the drag and drop operation between the Display panel and Execution panel. User can select table or column display panel then drops them into execution panel. This module displays that user selected item on the Execution panel. This module provides the drag and drop operation between the display panel and execution panel. So easily user can select the table or column and then drag that from display panel, then drop them into execution panel. After user has finished their selections, this module sends the information into business logic module. Then it collects the result from the business logic module. It stores the result in the local system and displays the result in result panel in the grid view form [13] [14].

Business Logic Module

This module mainly performs all the following operation in background.

Connection Module

This module interacts with interface module and gets the user's information. From that information, it finds the providers name for the database name. By using that provider name and other information (datasource, username, and password), It makes the connection string and then passes that string to Database Module. Database module sends back the connection. It maintains that connection along with the user information in the local system. This module also has capability to connect different database at the same time and maintain that information.

Table information module

This module gets the Database connection from the connection module and passes the connection and database name with database module. Then get backs the result of all table information of Database from Database Module. That result has the entire table and their column names and columns property. This result sends to the interface module to display result in the display panel.

Selected Table item

This module interacts with the Interface module collect the selected table item in execution panel. After it collects the information, it validates and removes the duplication. It stores that information in the local system. It finds the relationship among the selected table items and then stores the relationship information.

Query Generator

After the user finishes the selection in the interface module, The Query Generator Module is invoked by interface module. This query generator module gathers the selected table item along with filter conditions and the relationship from the selected table item module. it forms the query based on the filter condition and the relationships among the selected items. It sends the generated query to the Database Module. Get back the results of the generated query from the database module and sends the result to the interface module that displays the result on the display panel [15] [].

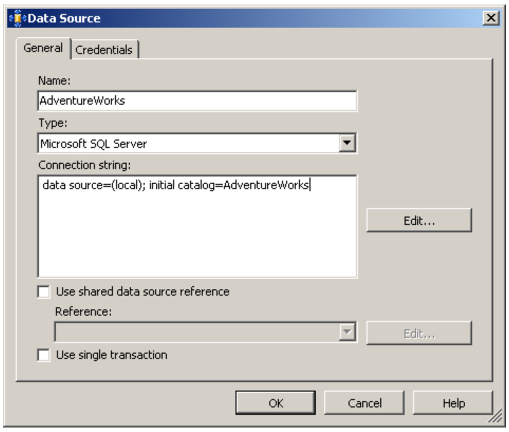
Database Module

This Database Module performs all database related operations. This has three different modules.

a. Make the Connection with Database.
b. Collection of Table information
c. Result Collection.

Make the Connection with Database

This module provides the functionality makes the connection to the database using connection string and returns the connection. The Business logic Module (4.2.1) uses this functionality to make a connection to the database and get connection. Figure 4 illustrates the Data source connection.



Data Source

General | Credentials

Name: AdventureWorks

Type: Microsoft SQL Server

Connection string: data source=(local); initial catalog=AdventureWorks

□ Use shared data source reference

Reference:

□ Use single transaction

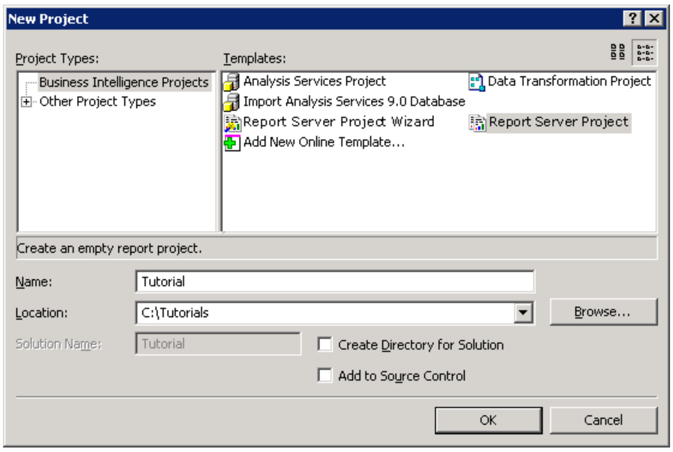
Figure 4: Data source connection.

Collection of Table information

Based on the DB and connection, this module forms the database related query for collecting entire table and their columns and column's properties of Database. This module is used by table information Module (4.2.2) of business logic module to gather entire DB information.

Result Collection

This module provides the functionality to execute query on the database and gets the result of the query from database. This module requires the connection and query string to execute the query. After query is generated by the query generator module, the query generator module invokes this module with that query and connection. This module executes the query and gets the result of the query from the DB and then this result sends back to the query generator module. After executing query it will be stored in a report as sown in figure B.



New Project

Project Types:
- Business Intelligence Projects
- Other Project Types

Templates:
- Analysis Services Project
- Import Analysis Services 9.0 Database
- Report Server Project Wizard
- Add New Online Template...
- Data Transformation Project
- Report Server Project

Create an empty report project.

Name: Tutorial

Location: C:\Tutorials

Solution Name: Tutorial

□ Create Directory for Solution

□ Add to Source Control

Figure 5: Create a new report project.

Implementation

Using our proposed framework, it could integrate many related database resources easily, the process can be accomplished using a single query, consider three different databases PIRPSD, PUBMED and DIPDB from Oracle, DB2 and Sybase respectively [18] [19]. The workflow can be described in figure 6.

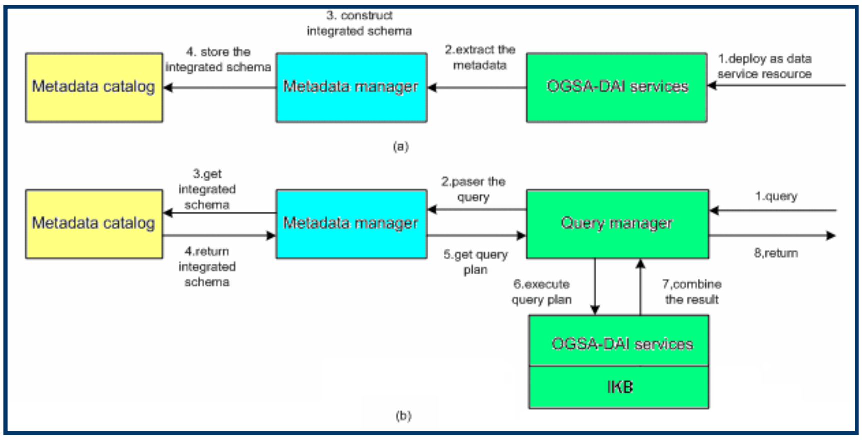
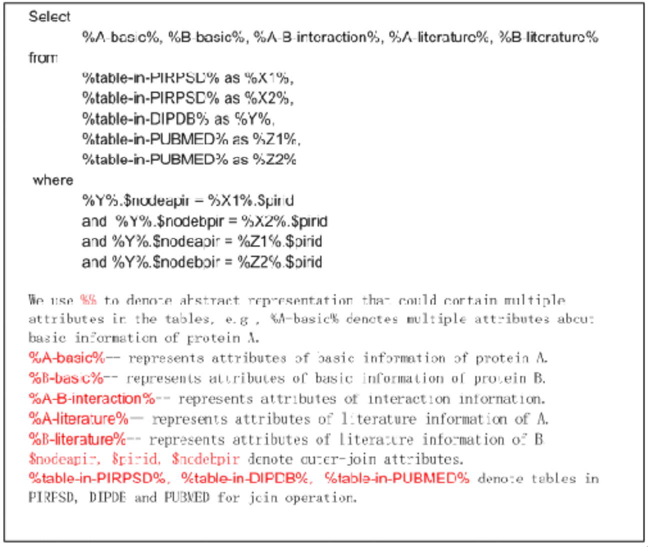


Figure 6: (a) Deploying database, (b) Query processing.

From Figure 6(a), we know that the deploying process firstly make the database be accessible from other machines as grid service, secondly update the integrated schema. After deploying databases according to workflow of Figure 3(a), researches can query the heterogeneous databases PIRPSD, PUBMED and DIPDB as local database [16] - [20], join operation is needed as follows to accomplish all jobs:



```
Select
        %A-basic%, %B-basic%, %A-B-interaction%, %A-literature%, %B-literature%
from
        %table-in-PIRPSD% as %X1%,
        %table-in-PIRPSD% as %X2%,
        %table-in-DIPDB% as %Y%,
        %table-in-PUBMED% as %Z1%,
        %table-in-PUBMED% as %Z2%
 where
        %Y%.$nodeapir = %X1%.$pirid
        and  %Y%.$nodebpir = %X2%.$pirid
        and %Y%.$nodeapir = %Z1%.$pirid
        and %Y%.$nodebpir = %Z2%.$pirid
```

We use %% to denote abstract representation that could contain multiple attributes in the tables, e.g , %A-basic% denotes multiple attributes about basic information of protein A.
%A-basic%-- represents attributes of basic information of protein A.
%B-basic%-- represents attributes of basic information of protein B.
%A-B-interaction%-- represents attributes of interaction information.
%A-literature%— represents attributes of literature information of A.
%B-literature%-- represents attributes of literature information of B
$nodeapir, $pirid, $nodebpir denote outer-join attributes.
%table-in-PIRPSD%, %table-in-DIPDB%, %table-in-PUBMED% denote tables in PIRPSD, DIPDB and PUBMED for join operation.

Figure 7: SQL for distributed query.

The query is submitted to query manager, metadata manager decompose the query to query plan according to the integrated schema stored in metadata catalog, query manager execute the query plan by coordinating the corresponding data services, transaction manager will be used in the process when concurrency occurs. The distributed query would take a long time to finish the outer join between tables in different database, then employing notification manager to inform the end of query is a favorable manner.

Conclusion

The IKBHDOA supports all kinds of databases. The user can easily retrieve and manipulate the information from the multiple databases without knowing the database related information. This kind of application is mostly help full in the organizations like hospital, agriculture, mechanical industries, etc. They need database knowledge people for writing the query and collecting the information from their database. With the help of this software, the user who just knows about project database data can gather all the information without knowing any technical details of Database.

References

[1] Teng Lv, Ping Yan (2007) “XML Functional Dependencies ”, IEEE Data, Privacy, and E-Commerce, 2007. ISDPE 2007.

[2] Zhang Zhenyou, Wang Honghui, Zhang Hao, "Research of Heterogeneous Database Integration Based on XML and JAVA Technology," eeee, pp.275-278, 2009 International Conference on E-Learning, E-Business, Enterprise Information Systems, and E-Government, 2009.

[3] Jin-Yong Tao, Qu-Feng Juan, Wang-Hui Juan, "Ontology-Based Research on Heterogeneous Database Semantic Integration Strategies," etcs, vol. 3, pp.477-480, ISBN: 978-0-7695-3987- 2010 Second International Workshop on Education Technology and Computer Science, 2010

[4] M. N. Alpdemir, A. Mukherjee, N.W. Paton, P.Watson, A. A. Fernandes, A. Gounaris, and J. Smith. “Service-based distributed querying on the grid”. In the Proceedings of the First International Conference on Service Oriented Computing, Springer, 15-18 December 2003, pp. 467-482.

[5] P. Ziegler, K.R Dittrich, “User-Specific Semantic Integration of Heterogeneous Data: The SIRUP Approach”, In First International IFIP Conference on Semantics of a Networked World (ICSNW 2004), volume 3226 of Lecture Notes in Computer Science, Springer, Paris, France, June 17- 19, 2004, pp. 44-64.

[6] Rahm, E, and P. A. Bernstein, "A Survey of Approaches to Automatic Schema Matching", VLDB Journal 10, 4, Dec. 2001, pp. 334-350.

[7] L. Zamboulis, H. Fan, K. Belhajjame, J. Siepen, A. Jones, N. Martin, A. Poulovassilis, S. Hubbard, S. M. Embury, N. W. Paton, “Data Access and

Integration in the ISPIDER Proteomics Grid", In Proc Data Integration in the Life Sciences 2006, July 2006.

[8] Cha, S.K. "Kaleidoscope: a cooperative menu-guided query interface (SQLversion)", IEEE Artificial Intelligence Applications, Vol.1. 2007.

[9] T. Landers and R. Rosenberg, "An Overview of Multibase ", in Distributed Databases, H.J. Schneider, Ed., North-Holland, The Netherlands, pp. 153-184. 2006.

[10] Y. Tohsato, T. Kosaka, S. Date, S. Shimojo and H. Matsuda, "Heterogeneous Database Federation Using Grid Technology for Drug Discovery Process", Grid Computing in Life Science: First International Life Science Grid Workshop, LSGRID 2004, Kanazawa, Japan, May 31-June 1, 2004.

[11] Krause, S. Laws, J. Magowan, N.W. Paton, D. Pearson, T. Sugden, P. Watson, and M. Westhead. "The Design and Implementation of Grid Database Services in OGSA-DAI", Concurrency and Computation: Practice and Experience, Vol17, Issue 2-4, pp. 357- 376, February 2005.

[12] R. Stevens, P. Baker, S. Bechhofer, G. Ng, A. Jacoby, N.W. Paton, C.A. Goble, "A. Brass: TAMBIS: Transparent Access to Multiple Bioinformatics Information Sources", 2004

[13] Sainan, Liu Caifeng, Liu Liming, Guan (2008) "A Storage Method for XML Document Based on Relational Database", IEEE Computer Science and Computational Technology, ISCSCT '08. Vol: 1. 2008.

[14] Kevin Loney, George Koch, "Oracle 9i the Complete Reference" TATA McGraw- Hill. 2007.

[15] The Garlic Project, URL: [http://www.almaden.ibm.com/cs/garlic/].

[16] The OBSERVER Project, URL: [http://siul02.si.ehu.es/OBSERVER/].

[17] The ISPIDER Project, URL: [http://www.ispider.man.ac.uk/].

[18] The OGSA-DAI: URL: [http://www.ogsadai.org.uk/]

[19] The OGSA-DQP Project, URL: [http://www.ogsadai.org.uk/about/ogsa-dqp/].

[20] The Spitfire Project, URL: [http://edgwp2.web.cern.ch/edgwp2/spitfire /index.html]

Authors Biography

Mr. G. Appasami was born in Pondicherry, India in 1980. He received his Master of Science degree in Mathematics, Master of Computer Applications degree and Master of Technology degree in Computer Science and Engineering from Pondicherry University, Pondicherry, India. Currently he is working as Assistant professor in the Department of Computer Science and Engineering, Dr. Pauls Engineering College, Villupuram, Tamil Nadu, Affiliated to Anna University of Technology, Chennai, India. His Area of interests includes image processing, database and web technology.

Mr. S. Karthikeyan was born in Madurai, India in 1984 He completed his master of Engineering in Computer Science and Engineering from Krishna engineering College, Coimbatore, Tamilnadu, India. He is currently working as Assistant professor in the Department of Information Technology, Bannariamman Institute of technology, Affiliated to Anna University of Technology, Coimbatore. His research area includes Database technology and Artificial intelligence.