School of Computer Sciences

Universiti Sains Malaysia

UNIVERSITI SAINS MALAYSIA

FINAL REPORT

DISTRIBUTED DATABASE MANAGEMENT SYSTEM

Project Number: 304/PKOMP/636031

Dr. Chan Huah Yong
Senior Lecturer
School of Computer Sciences
Universiti Sains Malaysia.

TABLE OF CONTENTS

1.0. INTRODUCTION

Data plays a fundamental role in all kinds of cross-organizational research and collaborations. These organizations can be collectively deemed to form virtual organization (VO) [1]. Data will exist in a variety of different formats, such as unstructured or multimedia files in file systems, structured collections stored in relational or XML databases [2], which can vary in volume and may be geographically distributed over a VO. Besides, some large projects such as LHC [3] will generate multiple terabyte-sized or even petabyte-size data. It is impossible to handle such large amounts of data within a single organization or institute. Addressing data access and integration across organizations is going to be one of the big challenges in setting up VOs. At present, Grid technologies are being developed to facilitate "coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations" [1]. The grid is "a system that coordinates distributed resources using standard, open, general-purpose protocols and interfaces to deliver nontrivial qualities of services." [4] The concept of VOs proposed by [1] can be regarded as a framework in which data access and integration have to be addressed.

In this project, we aim to implement a distributed database management system (DDBMS) and the type of the DDBMS will be both homogeneous and heterogeneous distributed DBMS. Two transparencies will be used in this project: distributed transparency and failure transparency. The user need not know the location of the data, data fragmentation and data replication in distributed transparency. In failure transparency, the system will provide the solution when server is crashed. For example, when this problem occurs, the first client which is trying to access to the database will become the new server and the old server will become normal client. As a result, it will provide a better service to the user.

Our project is using Open Grid Services Architecture Data Access and Integration

Web Service Resource Framework (OGSADAI-WSRF 2.2) as the middleware to develop the DDBMS. At the end of the project, the system will provide client and server services where users send service request such as create, retrieve, update and delete data resources from distributed computing databases. A web application is implemented to provide a user friendly environment for users.

1.1. Motivation

The motivation of building up the distributed database management system is based on the advantages that the project may gain. A distributed database management system can reflect an organization structure, in which, the database fragments are located in the departments they relate to. The system can achieve local autonomy where a department can control the data about them as they are the ones familiar with it. By using distributed database management system, it can improve the performance because a fault in one database system will only affect one fragment, instead of the entire database. In addition, the system can improve the performance as the data is located near the site of the greatest demand, and the database systems themselves are paralyzed, allowing load on the databases to be balanced among servers. It is considered economic to build a distributed database management system. It costs less to create a new work of smaller computer with the power of a single large computer. Furthermore, the system can be modified, added and removed from the distributed database without affecting other modules, which add the value and great achievement in modularity of the system.

1.2. Objectives

The objective is to provide a local mapping transparencies and able to do fragmentation or replication of data. This project is not going to use any third party distributed database management system but develop our very own version of distributed database management system.

We will extend the local database to support distributed database that is easily

implemented in applications such as student information systems or retailing information system.

1.3. System Description and Function

The main purpose of distributed database management system is to transparent data resource inside distributed computer system to users. The system will use Open Grid Services Architecture Data Access and Integration Web Services Resource Framework (OGSADAI-WSRF) as a middleware to manage Grid Data resources via web services. This system will provide client and server services where users send service request such as select, add, modify or delete data from distributed computing databases. In this project, we aim to develop a web application which will provide a user friendly environment for system user instead of using command line input.

This system consists of three basic modules – web application, database and OGSADAI-WSRF. The web application is to provide a user-friendly environment to user to use the system. The database will be provided by the client side from a grid computing environment, and will be used to test the web application of DDBMS. Exploration of OGSADAI-WSRF and deployment under user machines will be done in the project. As OGSADAI-WSRF 2.2 is the latest version of web services middleware for database usage, we plan to explore the tool in this project.

1.4. System Outcomes

At the end of the project, we manage to develop a distributed database management system which makes use of 3 personal computers: 1 as the server and 2 as the database storage. This system can do a basic data query and fragmentation on the database management system. Through out this project, we manage to have one bachelor degree student in human capital and export development, who is involved in the final year project and will complete his bachelor degree majoring in computer science this year.

2.0. RELATED WORK

2.1. DataMiningGrid

The Data Mining Tools and Services for Grid Computing Environment (DataMiningGrid) Consortium is developing tools and services for deploying data mining application on the grid. Future and emerging complex problem-solving environments are characterized by increasing amounts of digital data and rising demands for coordinated resource sharing across geographically dispersed sites. The next generation of grid technologies is promising to provide the necessary infrastructure facilitating seamless sharing of computing resources. Currently, there exists no coherent framework for developing and deploying data mining application on the grid. Hence, the DataMiningGrid project addresses this gab by developing generic and sector-independent data-mining tools and services for the grid.

The shift towards intrinsically distributed complex problem solving environments is prompting a range of new data mining research and development problems. These can be classified into the following broad challenges for the project:

Distributed data:

The data to be mined is stored in distributed computing environments on heterogeneous platforms. Both for technical and for organizational reasons it is impossible to bring all the data to a centralized place. Consequently, development of algorithms, tools, and services is required that facilitate the mining of distributed data.

Distributed operations:

In future more and more data mining operations and algorithms will be available on the grid. To facilitate seamless integration of these resources into distributed data mining systems for complex problem solving, novel algorithms, tools, grid services and other IT infrastructure need to be developed.

Massive data:

Development of algorithms for mining large, massive and high-dimensional data sets (out-of-memory, parallel, and distributed algorithms) is needed.

Complex data types:

Increasingly complex data sources, structures, and types (like natural language text, images, time series, multi-relational and object data types etc.) are emerging. Grid-enabled mining of such data will require the development of new methodologies, algorithms, tools, and grid services.

Data privacy, security, and governance:

Automated data mining in distributed environments raises serious issues in terms of data privacy, security, and governance. Grid-based data mining technology will need to address these issues.

User-friendliness:

Ultimately a system must hide technological complexity from the user. To facilitate this, new software, tools, and infrastructure development is needed in the areas of grid-supported workflow management, resource identification, allocation, and scheduling, and user interfaces.

In order to support the above listed challenges new technology and services are being developed. The DataMiningGrid project focuses mainly on the development of the components and services belonging to DataMiningGrid Technology Layer. Analysis services and Data Services are being developed which will make the use of different data mining algorithms (located in the layer below them) in distributed environment possible. Workflow Editor is being developed, which eases the hard task of integration and configuration of complex analysis tasks.

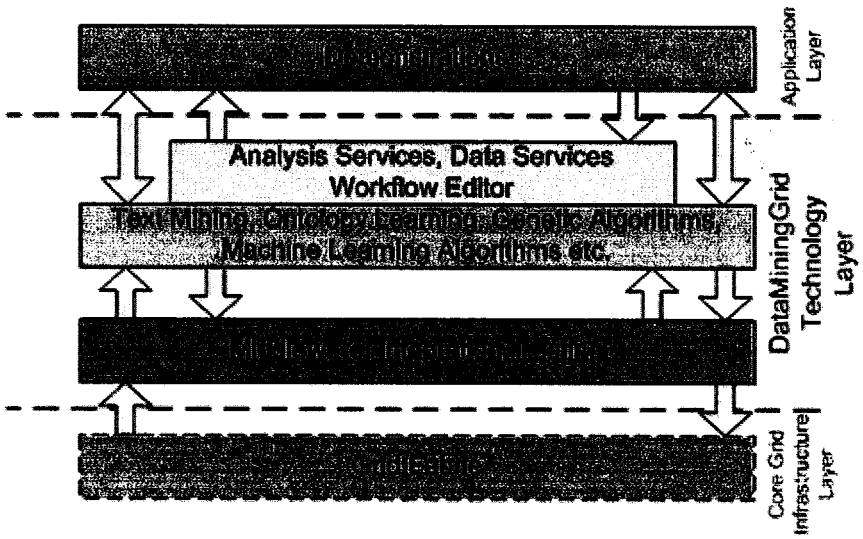

Figure 2.1: The DataMiningGrid Technology Layer

The application layer on the top of the stack represents a set of applications from a diverse range of domains and sectors that will be used to demonstrate the benefits of the developed technology.

2.2. Bio Grid

Data Grid Technology has been developed to gather and to securely process huge and various datasets and databases. In addition, a variety of databases are used collaboratively and a large amount of computational resources are needed to process data. BioGrid is aimed to develop a Computer Grid Technology that is able to satisfy all the above requirements of function. This technology can be used as the core technology to realize future Super Computer Network environment. Technically, peta grid technology is developed to process peta byte data by employing distributed computers of peta FLOPS capacity. Osaka University and other relevent institutions are in the progress of developing such a technology to meet the IT needs for its world class research activities in medicine and biology. The core institution carrying out this research project is the Cybermedia Center of Osaka University that is equipped with high capacity computational and network resources. Relevent research institutions and private enterprises also participate as peripheral research groups that are connected by high-speed network. Apart from research activities, the project also

includes peta grid technology business development and educational training in bioinformatics. The objectives of this project are two-fold: in silico pharmaceutical research and development based on Biochemistry, InformationScience, Mathematical Chemistry, Physics, Medicine and Pharmacology; and establishment of Super Computer Network core technology to be used in other research fields with similar needs.

A dramatic advancement in science and technology has made it possible to observe not only microscopic cells but also the brain mechanism using leading-edge, sophisticated analyzers. Additionally, new protein structures and gene information have been made clarified, which may hold clues to the inquiry into the essence of the biological life process and contribute to future medical treatment. This project has been entrusted with the mission of attaining the following three goals:

1. A series of analyzers deployed on a supercomputer network,
2. "Data grid technology", which will allow to achieve total and safe linkage and manipulation among various types of huge databases
3. "Computing grid technology", which may facilitate truly systematic linkage among the databases and data processing requiring ultra high-sped computing resources.

The sights on outcome from the project such as a vast volume of data, as well as the next-generation technology and know-how for computing will not only realize in silico pharmaceutical development and biodynamic elucidation, but also support the establishment of the infrastructure technology for the supercomputer network in other arenas requiring similar technology.

2.3. Metadata Catalog Service (MCS)

MCS is a metadata catalog service that stores descriptive information (metadata) about logical data items. MCS has been developed as part of the Grid Physics Network (GriPhyN) and NVO projects. The aim of these projects is to support large-scale scientific experiments. MCS is a standalone catalog that stores information about logical data items (e.g., files). It also allows users to aggregate the data items into collections MCS provides system-defined as well as user-defined attributes for logical items and collections. One distinguishing characteristic of MCS is that users can dynamically define and add metadata attributes. MCS can also provide the names of the user-defined attributes. As a result, different MCS instances can be created with alternative contents. MCS have been implemented to run on top of standard web services or on top of the OGSA-DAI grid service. In the latter case MCS leverages the OGSA-DAI's authentication capabilities to provide secure access to the metadata.

MCS may be used for storing and accessing metadata about logical files. A logical file uniquely identifies the content of a file. MCS can be used in conjunction with the Globus Replica Location Service (RLS) that locates the physical instances of logical files. Among the attributes of logical files are: file creator, creating timestamp, etc.

Figure 2.2 illustrates the simplest scenario for attribute-based data access via the MCS and the other components of the data grid, including the Replica Location Service and the particular storage system where the data resides.

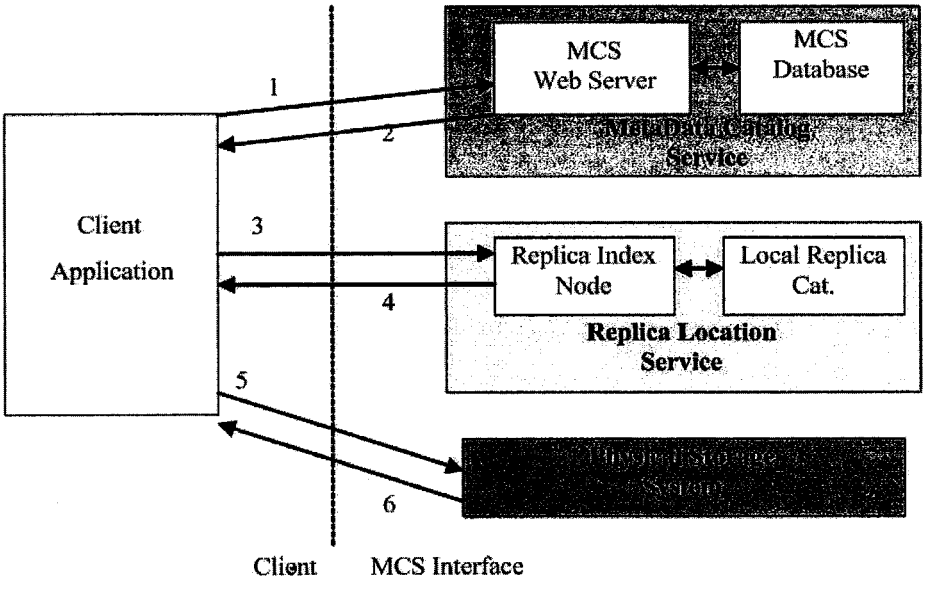

Figure 2.2: Scenario for attribute-based data access via MCS

A usage scenario of the Metadata Catalog Service:

1. The client application (or a request planner tasked with satisfying a client's request) queries the Metadata Service based on some attributes of the desired data.
2. The MCS returns the logical file names of one or more data items corresponding to those attributes.
3. The client application next queries the Replica Location Service with the logical file names.
4. The RLS returns the physical file names of the requested files to the client application.
5. The client application then contacts the physical storage system where the file resides (using GridFTP for example)
6. The file is returned by the physical storage system.

3.0. METHODOLOGY & BACKGROUND

3.1. Background

3.1.1. Distributed Database System

Distributed database is a database that is under the control of a central database management system in which storage device are not all attached to a common but distributed in different location. It may be stored in different and multiple computers located in the same physical location. It may be dispersed over a network of interconnected computers. A transaction can reach the data from one or more than one location of the distributed database. The collections of data can be distributed across multiple physical locations. The database is distributed into separate fragments where each of these fragments may be replicated. A distributed database system consists of loosely coupled sites that share no physical component. Database systems can run on each site and they are independent of each other. The transactions may access data at one or more sites.

The basic architecture of the distributed database includes a server and client. A database server is the software managing a database and a client is an application that request information from a server. Every computer in a system is a node. A node is a distributed database system which considered as a client, a server or both, depending on the situation.

There are a few models for the distributed database architecture as follows:

- **Fragmentation**

  Relationship between databases is partitioned into several fragments and stored in a distinct site. There are 3 types of fragmentation methods: Horizontal fragments, vertical fragments and mixed fragments.

- **Replication**

  The database system multiple copies of data, stored in different sites, for a faster retrieval and fault tolerance.

- **The combination of replication and fragmentation**

  Relation is partitioned into several fragments: system maintains several identical replicas of each such fragment.

There are 2 types of distributed database: Homogeneous distributed database and Heterogeneous distributed database. Homogeneous distributed database uses one database management system (DBMS). All sites have identical software. The sites are aware of each other and agree to cooperate in processing user requests. Each site surrenders part of its autonomy in terms of right to change schemas or software. It appears to user as a single system. In a heterogeneous distributed database, different sites may use different schemas and software. The sites may not be aware of each other and may provide only limited facilities for cooperation transaction processing.

The important considerations with a distributed database must be taken are the distribution is transparent and the transactions transparent. Users must be able to interact with the system as if it was one logical system. Each transaction must maintain database integrity across multiple databases. Transaction must also be divided into sub-transactions, each sub-transaction affecting one database system.

3.1.2. Distributed Database Management System (DDBMS)

Distributed Database Management System (DDBMS) is a software system that permits the management of the distributed database and makes the distribution transparent to the users. The architecture of the DDBMS can be designed as client-server systems or peer-to-peer. Nowadays, the design is more to client-server system. One or more servers manage the database and handles user queries that are passed on by the clients. The clients usually have limited database functionality and normally pass the SQL queries over the servers for processing. In peer-to-peer systems, each site has equal functionally for processing.

3.1.3. Web Service

Web service is self-contained, modular applications that can be described, published, located, and invoked over a network. In some object-oriented systems, some of the fundamental concepts in web services are encapsulation, message passing, dynamic binding and service description and querying. Web service is the notion that everything is a service, publishing an API for user by other service on the network and encapsulating implementation details.

In web service, there is several essential activities need to happen in a service-oriented environment:

1. Web service needs to be created; with its interfaces and invocation methods must be defined.
2. Web service needs to be published to one or more intranet or internet repositories for potential users to locate.
3. Web service needs to be located to be invoked by potential users.
4. Web service need to be invoked to be of any benefit.
5. Web service need to be unpublished when it is no longer available.

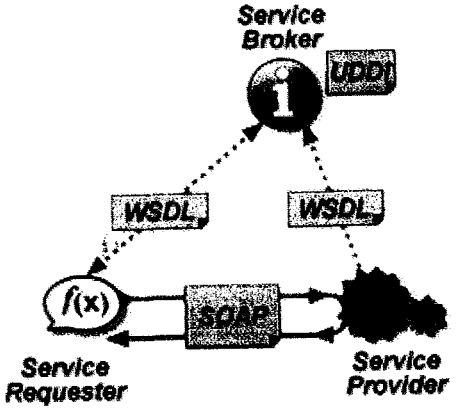

Figure 3.1: Web service overview

The common "core" specifications that are supplemented by others as the circumstances and choice of technology dictate are:

- **SOAP**

  An XML-based extensible message envelope format with "bindings" to underlie protocols.

- **WSDL**

  An XML format that allows service interfaces to be described, along with the details of their bindings to specific protocols. Typically used to generate server and client code, and for configuration.

- **UDDI**

  A protocol for publishing and discovering metadata about web services, to enable applications to find web services, either at design time or runtime.

3.2. Methodology

According to the design of the distributed database management system architecture, the project is split into 2 main components. The followings are the further description of the components:

- **Catalog Determination.** The definition in each of every catalog is using XML to represent specific information, fragmentation schema. The catalog consists:
  - The type of database
  - The attributes of the database
- **User Interface Application.** The general user interface will be implemented using Java language. Java programming provides the APIs which has the direct approach to data resources, mainly, JDBC for relational databases and the XMLDB API to connect to XML databases. The functionalities required for the system are:
  - Create Database
  - Retrieve Database
  - Update Database
  - Delete Database
  - Query Result Display

3.2.1. Development Methodology

In the development of this project, we apply the object oriented analysis and design (OOAD) methodology. The process of the development would be from building the system from the core up to the web application catalog. Databases are built to run the system testing at the end of the project. Managing the development of the project using the OOAD method, we embedded the research on the OGSADAI-WSRF concept, identify developer's roles, defining software and hardware needed for the project, planning the project schedule, and develop web application catalog and testing the system function. We employed a developer to develop this project during the implementation session.

3.2.2. Software Lifecycle

This project uses the Rapid Prototyping Lifecycle with Versioning (Spiral Model) for the development planning and scheduling because for new developed system, it needs to construct the prototype from time to time. As our expectation, the iterations include analysis, design, construction and testing were typically 6 months.

For phase I, we are going to determine client requirement, planning project schedule for whole system, determine system requirement specification, and analysis project risk.

For phase II, we will refine the methodology, design and implementation for the project. Unit coding & testing web application in manage grid data resource also will be done in this phase. At this phase, we will get our first prototype of the system.

For Phase III, a complete system should be done, tested with system integration function and preparing final documentation.

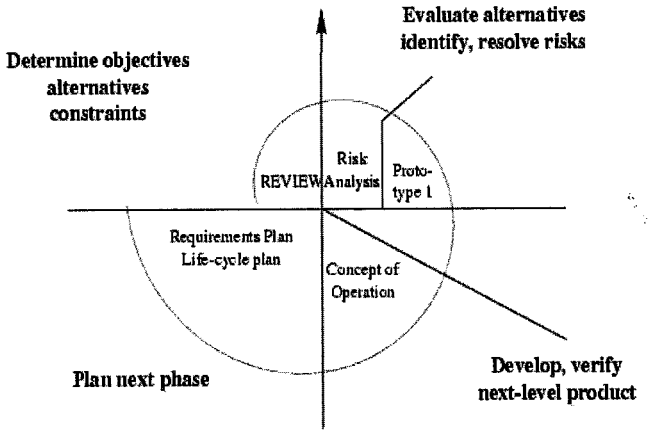

Figure 3.2: Rapid Prototyping Lifecycle with Software Versioning [14].

3.2.3. Gantt Chart

Year	2006												2007											
Month	Oct				Nov				Dec				Jan				Feb				Mac			
Week	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Project Activities																								
Phase 1																								
Software & Hardware Installation	✖	●																						
System Analysis & Design			✖	✖	●																			
Phase 2																								
Refining System Analysis & Design					✖	●																		
Deploying OGSADAI-WSRF						✖	✖	●																
Setting Up Databases								✖	●															
Unit coding & Web Application Testing							✖	✖	✖	●														
Prototype implementation										✖	●													
Phase 3																								
Design / Coding												✖	✖	✖	●									
Testing			✖													✖	✖	●						
Integration																			✖	●				
Revise overall system																					✖	●		
Final Documentation																							✖	●

● – Key Milestone

4.0. DESIGN & IMPLEMENTATION

The distributed database system that we are going to implement supports both homogeneous and heterogeneous distributed database, as the current distributed database system are mostly support homogeneous database type. At the first phase of the implementation, the system will support relational database, MySQL, and the XML database, Xindice. In this project, the tool that we will use as reference for implementation is the OGSADAI-WSRF, which is a middleware that provides data access and integration capabilities to a grid. The role of the OGSADAI-WSRF middleware is to present a unified programming model for application writers and mask out problems of heterogeneity and distribution [15]. OGSADAI-WSRF enables various homogeneous or heterogeneous data resources, such as relational databases, XML databases and even file system, to be accessed through a uniform web serviced based data access interface. The focus of this project is on the development catalog which is playing an important role in the system as they define the access requirements of the database.

In this session, we are discussing about the overview architecture of the OGSADAI-WSRF, which we used as the middleware to build the DDBMS in the project. After that, we will show the design and implementation which are based on the architecture of OGSADAI-WSRF.

4.1. Overview of OGSADAI-WSRF

Open Grid Services Architecture Data Access and Integration Web Service Resource Framework (OGSADAI-WSRF) is a middleware that allows databases such as MySQL, XML and PostgreSQL, to be accessed via web application[16]. OGASADAI-WSRF architecture has 4 main layers – data layer, business logic layer, presentation layer and client layer.

The data layer is the foundation of OGSADAI-WSRF. Data layer consists of different types of data resource such as MySQL, XML, PostgreSQL, and file. User may store data in distributed database but with OGSADAI-WSRF, the data resources are transparent to user like a centralize database.

The business logic layer contains data services resources to access multiple data resources. Data services resource is unique for each data resource. Data services resource is playing a role of "manager" in order to manage data resource. The "manager" receives performed documents from data service; it will perform the data resource access or data transportation functionality towards data resource and provides feedback as response documents.

The presentation layer provides web service for client to access data service resource. In OGSADAI, there are different components. One of the components is Web Service Resource Framework (WSRF), and the other one is Web Service Inter-operability (WSI). This system requires us to explore the business logic layer and presentation by communication between data service resource and data service to access data resource.

The client layer provides web service application programming interface (API) which compatibles with WSRF or WSI services. The further investigation in client layer is using higher level language to request service. Java, Python, Perl and PHP are

example of web service API. Within this project, web application such as catalog will be develop under this layer for user to manage distributed data resource.

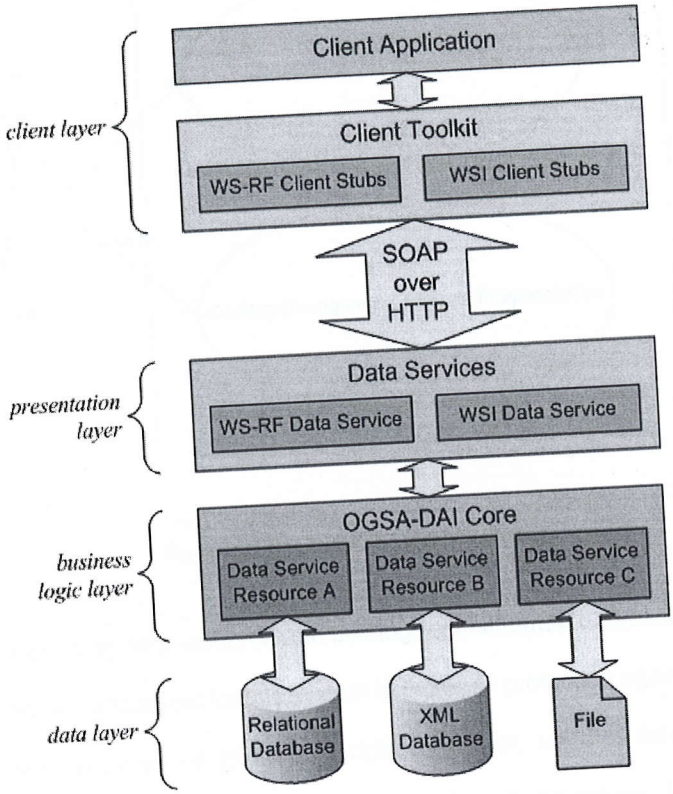

Figure 4.1: OGSADAI High-level Architecture [17].

4.2. Top Level Representation

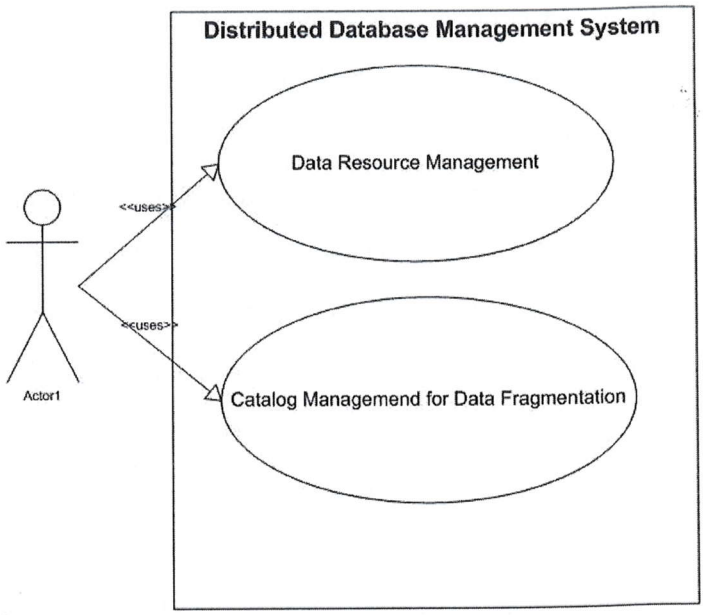

Figure 4.2: Top Level System Diagram

DDBMS is managing data resource and catalog for distributed database. For the security part will be the enhancement for this system to increase protection against non-authorities. Data resource management gives user right to search, update, delete and add data. These are the top level requirements for this project, build catalog for user to register fragmented or non-fragmented database. Web application catalog be able to retrieve data form distributed database without user send request to every database management system. Because, web application will retrieve data from every distributed database management system related to the user's request.

4.3. External Interface Requirements

4.3.1. User Interface in DDMBS

User will be able to manage data and catalog for fragmented table by using interface as high level language to create, retrieve, update and delete data or table in the database. User must input valid data in interface field and use button key to complete query function.

4.4. Internal Requirements for Web Application

The system user for this interface is only available for staff that is in-charge in managing the DDBMS. A web application data management system interface needs to be included. The web application will be enabling to use the searching function to search data from the DDBMS, to add data into the DDBMS. In addition, data can be deleted and also being updated from the DDBMS. When user clicks on the "Data Management Page", user can perform those query functions stated previously. The pre-condition of this use case is the user must give valid input to the field. Invalid input will make the system fail to perform the query function. If the query is successful, result will be displayed to the user.

Besides data management interface, catalog management need to be implemented. The catalog management gives user the right to manage fragmented data resource by add, delete, view and search table from distributed database. User clicks on the catalog management page and will lead the steps to manage the catalog by the basic function on the respective table. Hereby, user must have valid input to the system. The system cannot support with 2 similar table names. Hence, if a user wants to add new table name which is already an existing name, the old table name, which is similar, has to be deleted.

4.5. Internal Requirements for OGSADAI-WSRF

For the internal interface requirements to OGSADAI-WSRF, the system receives input commands from users, and then sends message to the middleware OGSADAI-WSRF which connects the related databases.

4.6. Internal Requirements for Database

In this interface for database, OGSADAI-WSRF will connect the database and the system will perform the query function. In this interface, OGSADAI-WSRF connects databases via driver and system performs command functions to the databases. If and only if OGSADAI-WSRF connected to the databases, the system can perform query command function.

4.7. Command Reference – User Interface

The main page shows all the available tables form deployed distributed database under OGSADAI-WSRF. User can click on tables available at left hand side to perform query. This link will lead user to search page to search data form these table.

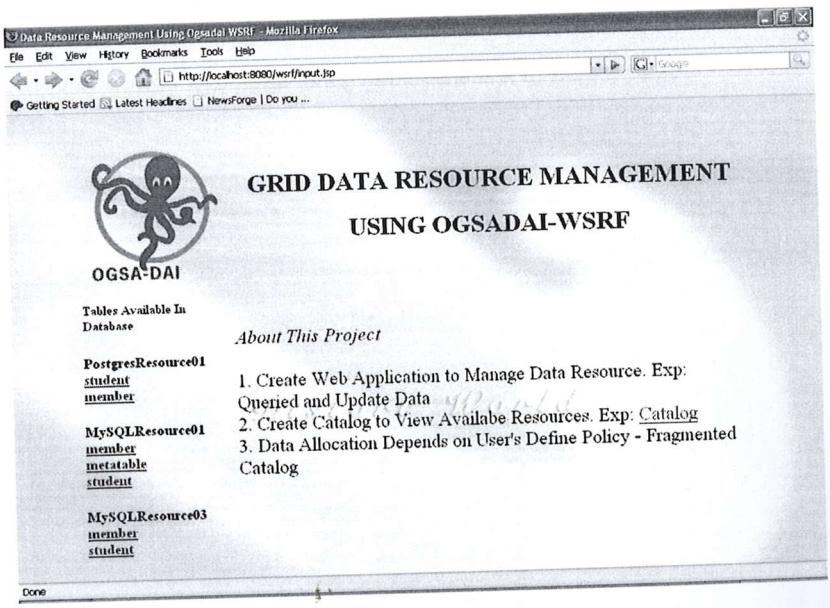

Figure 4.3: Home Page of the project.

Check the check box for data user want to search and key in values into the textbox. After that press "Search Button" to Proceed or "Clear button" to reset all the value. Example at below show that user want to search data from table name student with year=1. At the top page, there are 4 link which, HOME link is go to the home page, SEARCH page will link to data searching page, INSERT will link to insert data page and CATALOG will link to catalog page.

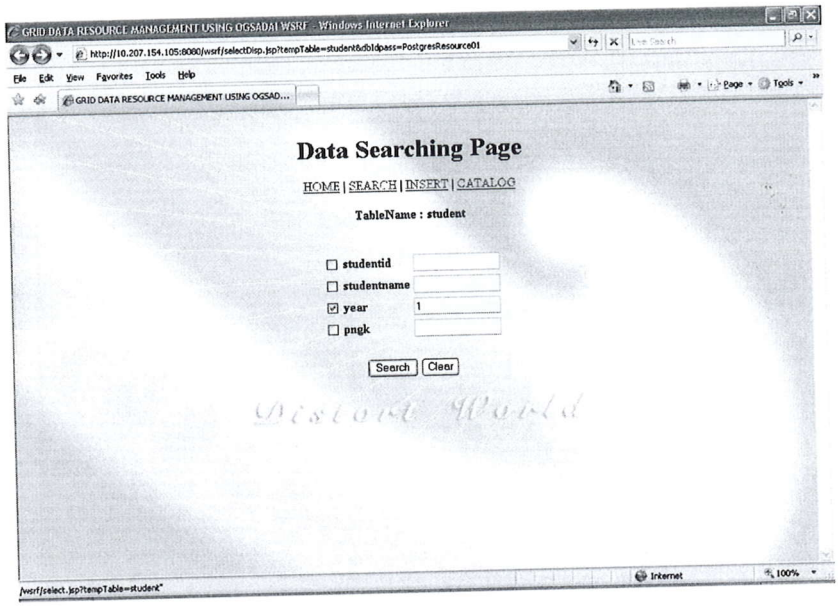

Figure 4.4: Data Searching Page

Now User can use the icon button to modify or delete that value. Modify icon will link to modify current row page while delete will link to delete page.

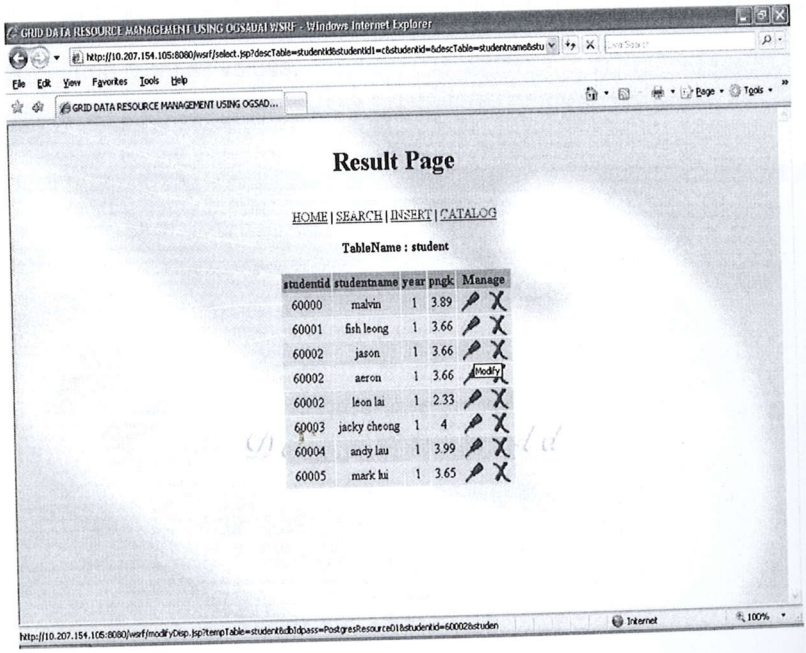

Figure 4.5: Result Display Page

If user click on delete icon, a popup confirm box will ask user to reconfirm delete particular row. If user want to proceed, press ok else press cancel button.

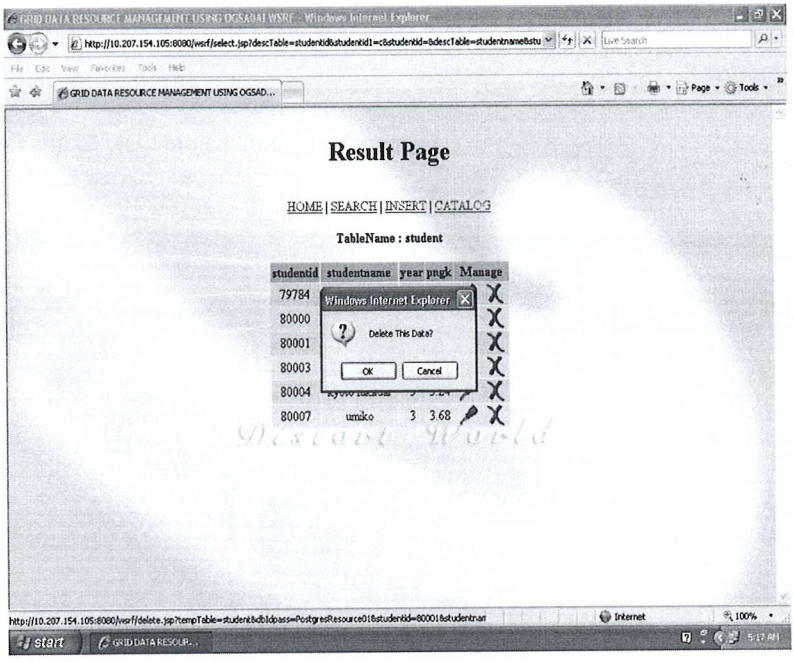

Figure 4.6: Delete Confirmation Pop-Up Box

INSERT link will allow user to logon Insert Data Page. User can insert data to that particular table. Submit button will help user to insert current data into database while clear button reset all values.

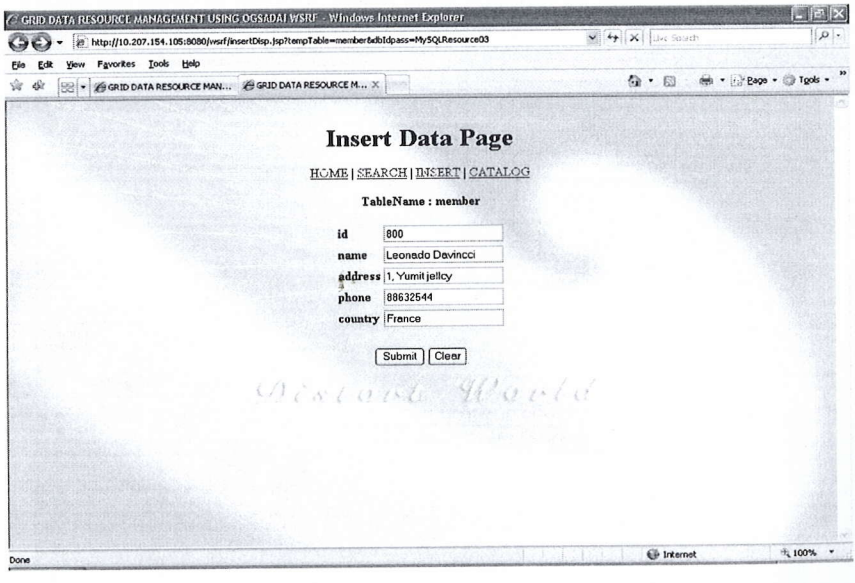

Figure 4.7: Insert Data Page

CATALOG link will link to catalog page which user can view all connected distributed databases and the info of databases. User may click the link in the Tables available column to view tables' further information.

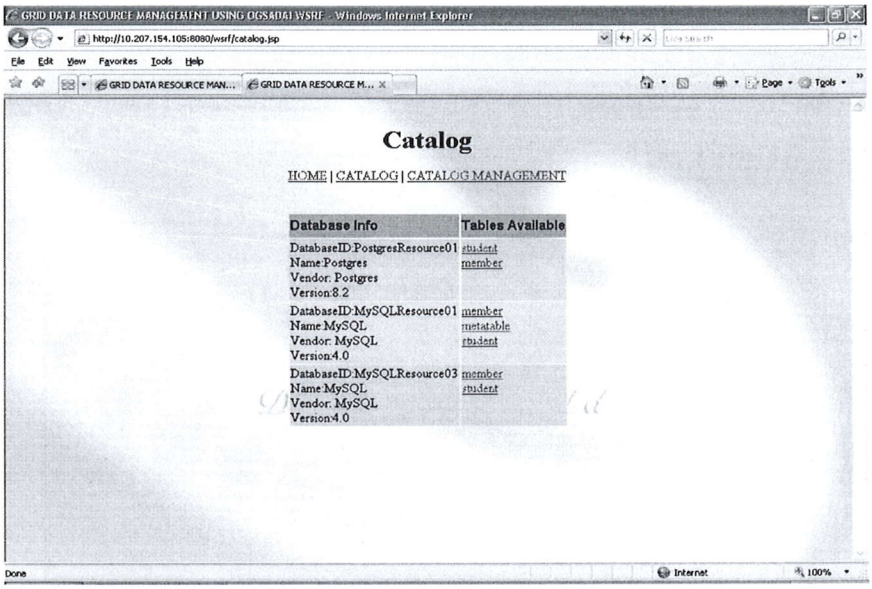

Figure 4.8: Catalog Page

Now we click the member link for DatabaseID:PostgresqlResource01. The button at bottom will show all the data inside table member for database PostgresResource01.

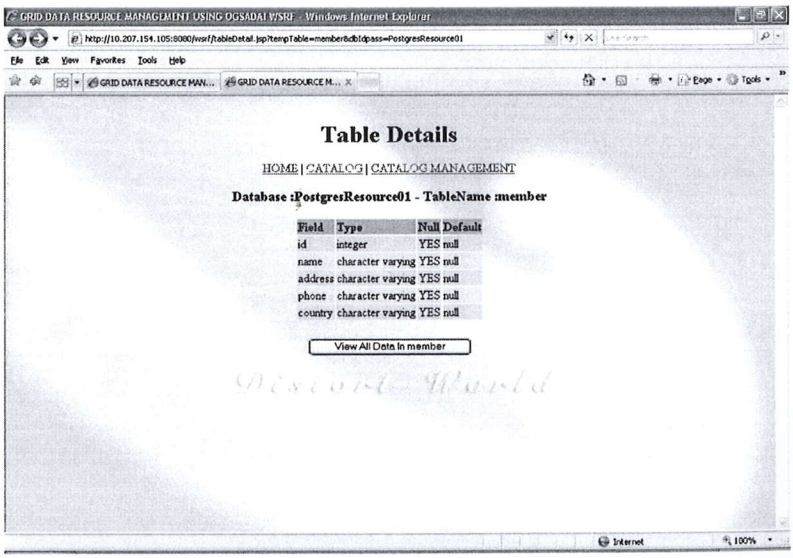

Figure 4.9: View Table Detail Page

CATALOG MANAGEMENT link will allow user to add, drop, alter table using sql query command. The drop down list allow user to choose to perform the query to which database. The submit query button will proceed to execute the query command.

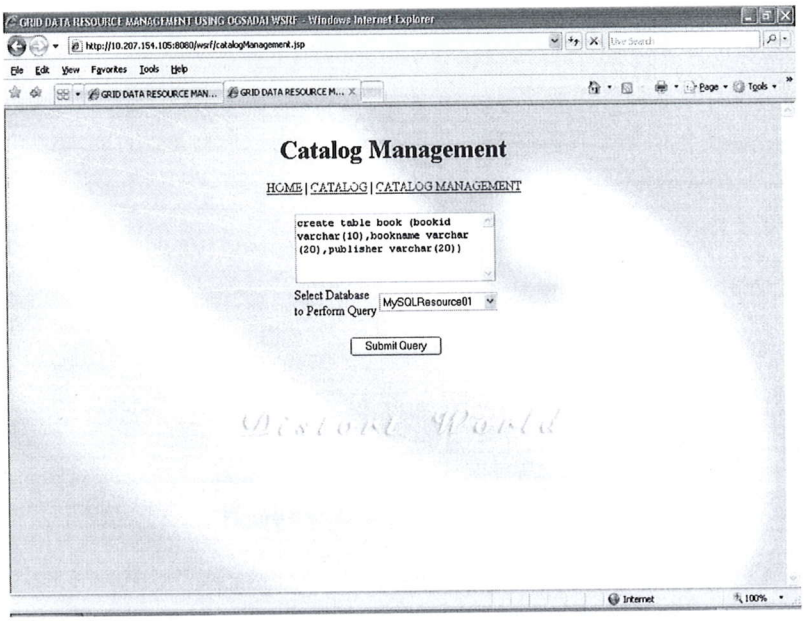

Figure 4.10: Catalog Management Page

Table metatable is a fragmented table catalog which means that this system will follow the user policy to perform data query. User may define the policy by using insert data page and modify it by using modify page.

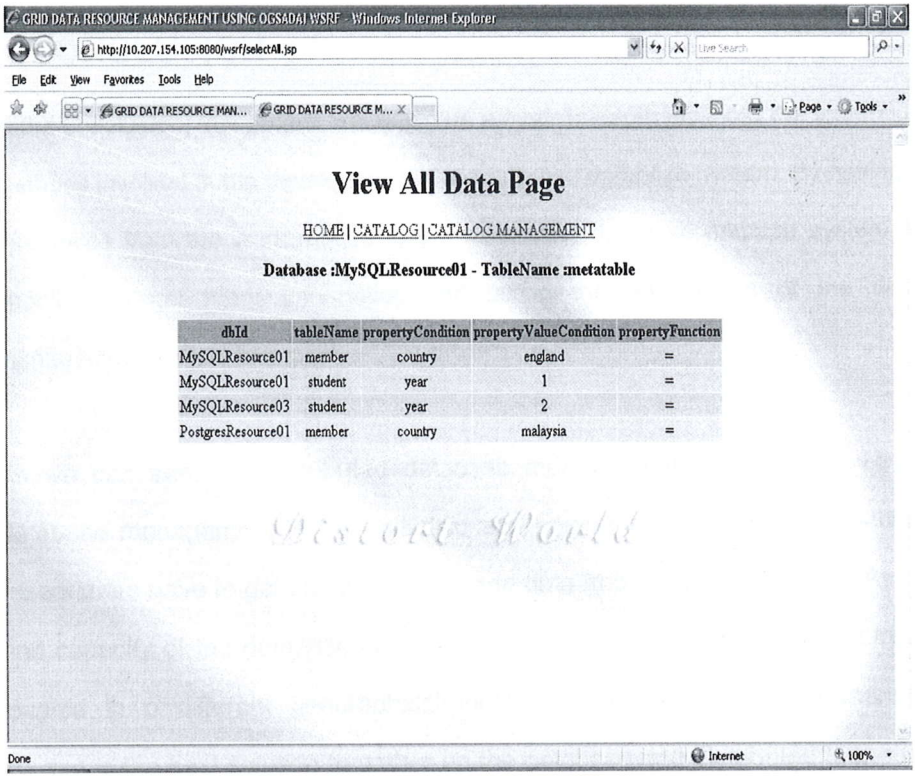

Figure 4.11: Data Fragmentation Catalog

dbId = Database id is the databases register with ogsadai-wsrf

tableName = Table inside the particular database

propertyCondition = Table attribute/field for value in tableName

propertyValueCondition = The value for user to define how the table fragmented

propertyFunction = "=" means the table will fragmented by propertyCondition=propertyValueCondition (this system currently only support "=" function, "< , >" function may add on in the future.)

If user wants to set up new database server, user must specify a database with id MySQLResource01 and setup a new metatable table to enable data allocation function.

5.0. CONCLUSION

In the conclusion, this project has given us a more in-depth look into the techniques and methods involved in the development of a distributed database system. By having related references from the work related to the distributed database manage system, we are enabling to get more information and taking the advantages of the distributed management system to implement the project.

As we can see, the distributed database management system is a very useful database management in the scientific and engineering field as the specialists or researchers have to get the information and data from all over the world. As the size and capacity of the database is getting greater, in addition, the physical location is located in a different geographical location, distributed database management system is the best solution to gather up the database and distributed to the users. It enhances the speed in accessing the database as users can retrieve or use the data they want from the nearest location of the data distributed.

Finally, despite a few shortcomings, the system can be successful in providing the basic level of distribution transparency and failure transparency as a functional distributed database system.

REFERENCES

[1] Foster, I., Kesselman, C. and Tuecke, S. *The Anatomy of the Grid: Enabling Scalable Virtual Organization*, International Journal of Supercomputer Applications 15(3), 200-222, 2001.

[2] The XML:DB Project. "*XML:DB Database API Working Draft*". http://xmldb-org.sourceforge.net//. Technical Report, 2001.

[3] The Large Hadron Collider Project: http://lhc-new-homepage.web.sern.ch/lhc-net-homepage/

[4] Ian Foster, Carl Kesselman, "Concepts and Architecture" *The Grid: Blueprint for a New Computing Infrastructure*, (Second Edition) Morgan Kaufmann, 2004. P46.

[5] Silberschatz, Korth and Sudarshan, "Distributed Database" *Database System Concepts*, (Fifth Edition) McGraw-Hill, 2005.

[6] Wikipedia, the Free Encyclopedia. "*Distributed Database*". http://en.wikipedia.org/wiki/Distributed_database. Article, 2006.

[7] The OGSA-DAI Project: http://www.ogsadai.org.uk/

[8] DataMiningGrid Project: http://www.datamininggrid.org/

[9] The BioGrid Project: http://www.biogrid.jp/

[10] The Metadata Catalog Service (MCS): http://www.isi.edu/~deelman/MCS

[11] Understanding Web Services, http://www.webopedia.com/DidYouKnow/Computer_Science/2005/web_services.asp, February 2007.

[12] Web Services, http://en.wikipedia.org/wiki/Web_service, February 2007.

[13] Web Services Architecture Overview, http://www-128.ibm.com/developerworks/library/w-ovr/, February 2007.

[14] Spiral Model: http://www.cs.odu.edu/~zeil/cs451/Lectures/01overview/process3/process3_htsu1.html, 14 Oct 2006.

[15] Blair, G..S., Coulson, G.., Robin, P. and M. Papathomas, *An Architecture for Next Generation Middleware*, Proc. Middleware '98, The Lake Distinct, England, November

1998.17 http://citeseer.ist.psu.edu/blair98architecture.html

[16] The University of Edinburgh, "*What is OGSA-DAI*": http://www.ogsadai.org.uk/about/ogsa-dai/, 2005-2006, 13 Oct 2006.

[17] The University of Edinburgh, "*OGSA-DAI Architecture*": http://www.ogsadai.org.uk/documentation/ogsadai-wsrf-2.2/doc/background/architecture.html , 2002-2006, 13 Oct 2006.