Distributed Heterogeneous Relational Data Warehouse In A Grid Environment

Saima Iqbal, Julian J. Bunn, Harvey B. Newman
California Institute of Technology, Pasadena, CA 91125, USA

This paper examines how a "Distributed Heterogeneous Relational Data Warehouse" can be integrated in a Grid environment that will provide physicists with efficient access to large and small object collections drawn from databases at multiple sites. This paper investigates the requirements of Grid-enabling such a warehouse, and explores how these requirements may be met by extensions to existing Grid middleware. We present initial results obtained with a working prototype warehouse of this kind using both SQLServer and Oracle9i, where a Grid-enabled web-services interface makes it easier for web-applications to access the distributed contents of the databases securely. Based on the success of the prototype, we proposes a framework for using heterogeneous relational data warehouse through the web-service interface and create a single "Virtual Database System" for users. The ability to transparently access data in this way, as shown in prototype, is likely to be a very powerful facility for HENP and other grid users wishing to collate and analyze information distributed over Grid.

1. INTRODUCTION

Data warehouse tend to push the limits of database storage capabilities and performance. Physically storing and administrating the data warehouse as a distributed database helps in the storage and access of large and small pieces of datasets.

Distributed Heterogeneous Relational Data Warehouse (DHRD) is a data warehouse technology. This can use to implement the storage and access of data from warehouse to multiple sites of relational databases like MS-SQL and Oracle. This could reduce the need of massive central computing resources, network delays and provides the transparent local datasets access to the clients [1].

The DHRD would involve different platforms, different administrating policies and very likely geographically distributed DHRD clients. The Grid provides a platform that potentially enables a systematic approach for this type of heterogeneous system.

This paper proposes a Web Services framework on the basis of Grid Services [2] to integrate DHRD with the Grid environment as a grid-enabled web services. This could enable the access of DHRD in distributed environment of various platforms.

This framework implements Grid Database Service Specification [3]. In addition to these specifications, it also provides a JAX-RPC [4] client to access DHRD across the Grid, MonALISA [5] integration to find optimal network resource path for the connection with DHRD and a private UDDI [6] registry server using Xindice for the repository.

A prototype builds on the proposed framework. The first test of the prototype addresses two questions: how a client can find the location of a database for a required dataset? How a UDDI registry server can be used for the registry and repository purpose of Grid Services based DHRD services.

2. GRID SERVICES

Grid Service is a Web Service that conforms to a set of conventions (interfaces and behaviors) that define how a client interacts with services available across the Grid [7] as shown in figure 1 and 2.

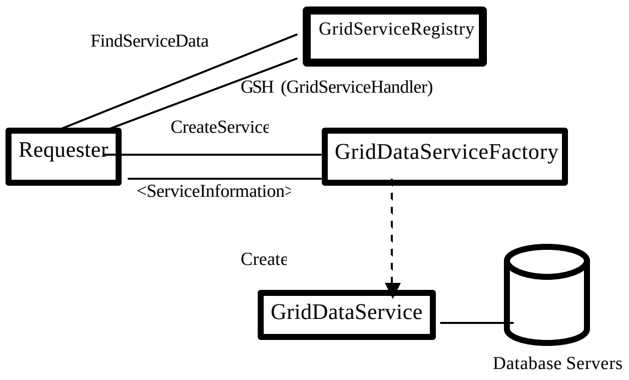


Figure 1: Creating a Grid Data Service

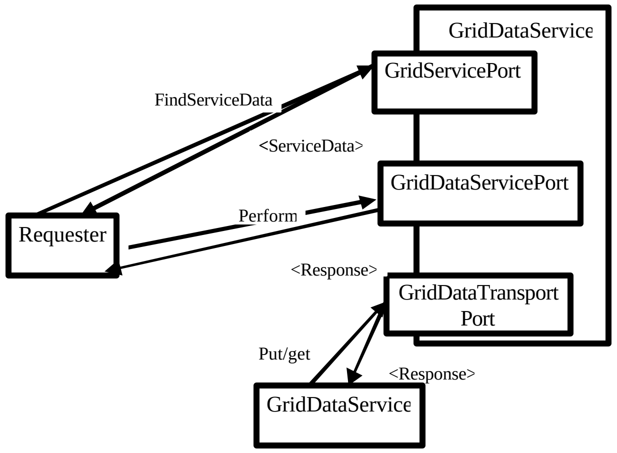


Figure 2: Requester Using Grid Data Service

The Open Grid Service Architecture (OGSA) [8] extends Web Services with consistent interfaces for creating, managing and exchanging information among Grid Services, which are dynamic computational artifacts casts as Web Services. Both the Web and Grid Services communities stand to benefit from provision of consistent, agreed service interfaces to database management systems (DBMS). Such interfaces requires to support the description and use of database systems (DBS) using Web Services standards, taking account of the design conventions and mandatory features of Grid Services.

3. WEB SERVICES

"*Web Services are modular software components wrapped inside a specific set of Internet communication protocols and that can be run over the internet*".

At the heart, Web Services architecture is the need of program-to-program communications and it is also the requirement of Grid Services. Key roles of the Web Services architecture are: a service provider, a service registry and a service requestor. Together they perform three operations of publish, find and bind on Web Services shown in figure 3. These operations are performed with the help of three vital components of Web services: Simple Object Access Protocol (SOAP), Web Services Description Language (WSDL) [9] and Universal Description Discovery and Integration (UDDI) technology.

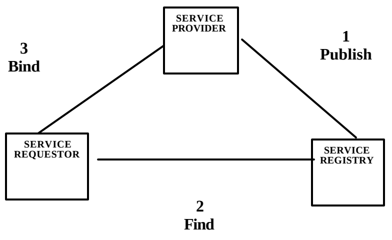


Figure 3: Web Services architecture

4. INTEGRATION OF DHRD WITH THE GRID

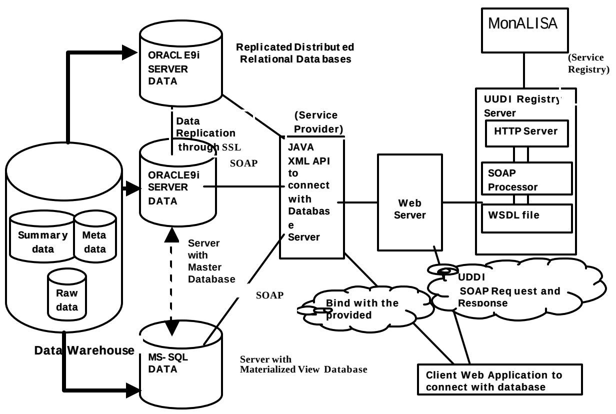

MonALISA
(Service Registry)
ORACL E9i SERVER DATA
Replicated Distribut ed Relational Data bases
Data Replication through SSL
SOAP
(Service Provider)
JAVA XML API to connect with Databas e Server
UUDI Registry Server
HTTP Server
SOAP Processor
WSDL file
Web Server
ORACLE9i SERVER DATA
Summary data
Meta data
Raw data
Server with Master Database
SOAP
Bind with the provided
UDDI SOAP Req uest and Response
Data Warehouse
MS- SQL DATA
Server with Materialized View Database
Client Web Application to connect with database

Figure 4: DHRD Web Services Framework for the Grid

Neither current Grid software nor existing Relational Database Management System (RDBMS) are able to fully support the integration of DHRD with the Grid. The proposed framework shown in figure 4 examines how this integration can achieve.

The framework proposes a web services driven approach. Each RDBMS involves in the distributed architecture of DHRD offer a set of services covering: the data type definitions that describe the data to be exchanged, the definition of the data to be sent to or from the service and the database operations supported by the service. Individual operations offered by these services would be standardized to increase portability and where possible reduce the effort required to build applications that interact with multiple relational databases. This would be done by adding code to map the operation interface offered by a service to the vendor specific interface beneath. However, it is impossible to standardize all services: for example different database paradigms support different types of query languages. Even within relational databases there are variations and these can not all be reconciled into a standard query language.

One advantage of the web service driven approach is however that each RDBMS made available on the Grid can provide a metadata service in form of WSDL that gives information on the range of services and operations that it supports. This would give application developers the information required to exploit whatever facilities are available.

In this framework, distributed databases of DHRD register themselves in UDDI registry server as a web service. The client, a service requester, sends a request for a required dataset to the web server as an http request. On the basis of the information provided by the request of the client, server-classes query the UDDI registry server. This query returns the endpoint of the required database. Here, datasets replicates from data warehouse to the distributed heterogeneous relational databases. Due to the replication of datasets there is a possibility to receive more then one endpoints for the required dataset. So, web server would also require querying the MonALISA to find the optimal network path to connect with required database.

After having the required service endpoint and optimal network path information client application binds to the required database of DHRD to access the datasets and for other database operations.

5. PROTOTYPE TEST

5.1.Technologies Employed

For this prototype JAX-RPC [4], stands for Java API for XML-based RPC, is used to create Web services and client that use Remote Procedure Calls (RPC) and XML. JAXR [10], Java API for XML registries, used to register and discover the services. Java Web Services Developer Pack (JWDP [11]) registry server 1.0_02, implements version 2.00 of UDDI and use Xindice database for the repository of the registry data. Xrpcc [12] tool to generate WSDL. Apache web server [13] and Tomcat servlet engine.

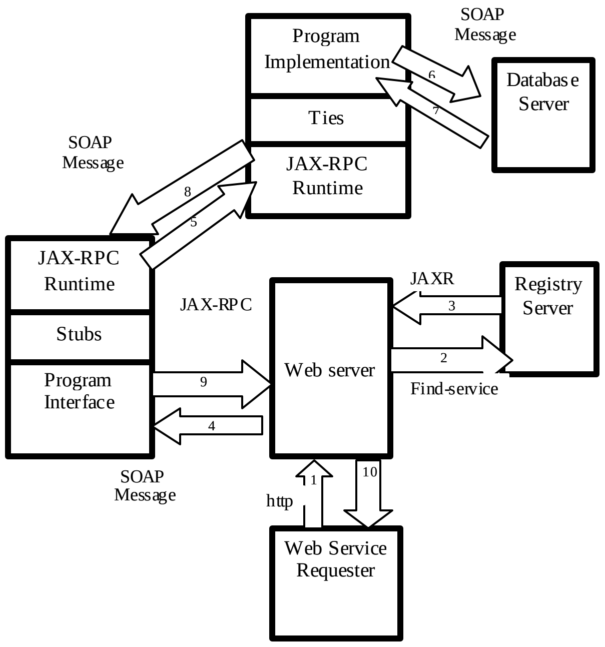


Figure 5: DHRD Services prototype at runtime

5.2. Working

Figure 5 shows a simplified view of the DHRD service after it has been deployed. After receiving the client's http request, web server using JAXR, sends a query to the UDDI registry server. Search for the required service (read WSDL) that supports JAX-RPC. After having the URL of **endpoint,** implementation class of the client program on web server invokes a method on the **stub**. The stub invokes the routines in the **JAX-RPC runtime system**. The runtime system converts the remote method call into a **SOAP** message as an **HTTP** request. When the server receives the **HTTP** request, the **JAX-RPC runtime system** extracts the **SOAP** message from the request and then translates it into the method call. The **JAX-RPC runtime system** invokes the method on the **tie** object. The **tie** object calls method on the implementation of the **DHRD service**. The run time system on the server converts the method's response into SOAP message and then transmits the message back to the client as an **HTTP** response. On the client side the **JAX-RPC runtime system extracts the SOAP message from the HTTP response and then translates** it into a method response for the client program.

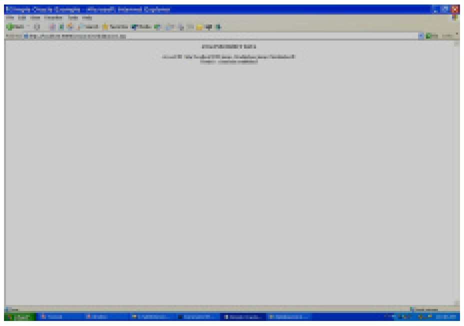
5.3.Screen Shots of Prototype

WEB SERVICE TO ACCESS ANALYSIS OBJECT DATA FROM [illegible]

[illegible]

[illegible]

Figure 6: JSP page to locate service


[illegible]

[illegible]

Figure 7: Response of the request: endpoint URL and database connection result

5.4.Prototype Result

The client of DHRD service receives the URI of the required database endpoint and the result of database connection. Here client just make a simple http call and receives the required result. Entire complex program-to-program communication of grid services based DHRD web services framework remains transparent for client application.

It concludes from the result of the prototype that DHRD can make accessible from different platforms with the help of Grid enabled Web Services.

6. FUTURE WORK

Future development of this prototype includes the integration of MonALISA (Grid Monitoring Tool). Further explore the WSDL and UDDI features for the solution of DHRD Services description and registry. Develop an API to integrate this DHRD Services prototype into the Globus toolkit.

Acknowledgments

This work is supported by CALTECH (California Institute of Technology), USA.

References

[1] Saima Iqbal. CMS IN 2002/002
Technical report on, Evaluation of Oracle9i to manage CMS event store: Oracle architecture to store Petabyte of data.
[2] I. Foster, C. Kesselman, Jeffrey M. Nick, S. Tuecke
The Physiology of The Grid: An Open grid Services Architecture for Distributed Systems Integration. 2002
[3] Susan Malaika, et. al
GGF Database Access and Integration Services Working Group. Grid Database Service Specification. 16th Feb. 2003
[4] Java™ API for XML-based RPC (JAX-RPC) 1.0_01 FCS Release Notes. 13th Sept. 2002
[5] MonALISA: http://monalisa.cacr.caltech.edu/
[6] UDDI Version 2.0 Data Structure Reference: UDDI Open Draft specification 8th June 2001
[7] GWD-R (draft-ggf-ogsi-gridservice-23) Open Grid Services Infrastructure (OGSI)
http://www.ggf.org/ogsi-wg
[8] I. Foster, C. Kesselman, Jeffrey M. Nick, S. Tuecke
The Physiology of The Grid: An Open grid Services Architecture for Distributed Systems, Globus Project 2002
[9] Web Services Description Language (WSDL) 1.1
W3C Note 15 March 2001
http://www.w3.org/TR/2001/NOTE-wsdl-20010315
[10] Java™ API for XML Registries (JAXR) 1.0_02 Release Notes
[11] Java™ Web Services Developer Pack 1.0_01 Release Notes
[12] Xrpcc:
http://java.sun.com/webservices/docs/ea2/tutorial/doc/JAXRPCxrpcc.html
[13] Apache Tomcat 4.1 for Java™ Web Services Developer Pack 1.0_01 Release Notes