ISSN:0975-9646

SCM: SCalable Middleware System for Heterogeneous Distributed Databases

Darshana Shimpi, Prof. Mrs. Sangita Chaudhri

Computer Engineering Department, Mumbai university
A.C. Patil College of engineering, Kharghar, Navi Mumbai, India

***Abstract-M*iddleware is an essential component for any type of distributed environment and network application. There are different middleware systems (like SQMD, DisCo, MOCHA, Open-Gate, OGSA-DAI, OGSA-DQP) which gives good performance, but most of them lack integrity constraints and scalability. Data integration using traditional database system or database gateway are not feasible or sufficient. Database middleware systems are used to integrate heterogeneous data sources distributed over network. There is a need to provide a mechanism which provides uniform view and access interface for heterogeneous data sets. Our SCM system gives design of scalable and efficient database middleware system for heterogeneous distributed databases with the characteristics of reliability, scalability and high performance. This middleware system has wrapper, integrator and query processor. It uses wrapper when integration of heterogeneous databases is required. We present the architecture of the scalable middleware system, the ideas behind it, the experiment and its result. The results of this middleware system provides flexible, scalable and efficient framework for heterogeneous distributed databases.**

Keywords-Heterogeneous Distributed databases, wrapper, middleware, integrator

I. INTRODUCTION

The intermediate software layer between application, operating system and communication system is known as middleware. Middleware migrate applications between client and server and also provide communication across heterogeneous platforms [3]. Middleware provides a uniform interface between a query and multiple, distributed databases. Services provided by middleware include enterprise application integration, data integration, message oriented middleware (MOM), object request broker (ORB), and the enterprise service bus. In distributed database system there are various loosely coupled sites that share no physical components [2]. In distributed databases, distribution of data is transparent to the user.

Middleware has some important functions as distribution transparency, uniform interface, hiding heterogeneity and use of some common services. Middleware is used to hide the distribution of data across multiple sites and it provides an uniform view to the user. In distributed database system, the system is with various operating system and hardware components. Middleware hides this heterogeneity. By hiding heterogeneity, middleware provides uniform view to the application developer. So these applications can be easily reused and ported. To perform various general purpose functions, middleware provides common services to the user as data integration, ORB etc. To make application development easy middleware provides common programming abstractions, also the middleware is used for masking the heterogeneity, the distribution of the underlying hardware and operating systems, and hide low-level programming details [4]. Middleware is divided into 5 categories according to the communications model adopted into middleware. They are database middleware, remote procedure call (RPC), transaction processing monitor (TP monitor), object request broker (ORB) and message oriented middleware (MOM).

Database-oriented middleware provides a number of important benefits as, an interface to an application, the ability to convert the application language into something understandable by the target database (e.g. SQL), the ability to send a query to a database over a network, the ability to process a query on the target database. In RPC the body of the procedure resides on a remote host and can be called the same way as a local procedure. Using RPC calls are sent to another machine across a distributed environment [1]. Procedural middleware supports remote procedure calls (RPC) and communications can be made by using primitives similar to local procedure calls. TP monitor simplifies the construction of a transactional distributed system. TP monitor is placed between the client program and the databases, it shows databases updated properly. It co-ordinates and monitor transactions across multiple data sources. TP monitor enhance the performance, reliability of server side systems [1]. ORB is the program that acts as a broker between a client request for a service from a distributed object or component and the completion of that request. Distributed object systems are CORBA, DCOM and EJB. Message Oriented Middleware uses messaging provider to integrate messaging operations. The main components of a MOM system are clients, messages, and the MOM provider. They provide an API and administrative tools. IBM MQSeries, Sun Java Message Queue are two examples of this category.

In some of the existing middleware systems like SQMD [5], OGSA-DAI[6], Open-Gate[7], MOCHA[8], DiSco[9], OGSA-DQP[10] translation of data items to the global schema is performed by either a wrapper or database gateway. All these systems are having different components in their architecture with certain features, advantages and disadvantages. These systems works on different homogeneous or heterogeneous datasets like object oriented, xml or relational databases in restricted application environment.

MOCHA and Open-Gate middleware systems are based on the general purpose database middleware systems, while other systems like SQMD, OGSA-DAI and OGSA-DQP are used for special purpose applications. For example, SQMD middleware system makes use of virtual private servers. OGSA-DQP can be used in any Grid application that aims to integrate and analyse structured data collections. DiSco is a mediator based middleware system. Mediators accept queries and transform them into subqueries that are distributed to databases. In DiSco scalability is achieved, but there is integration problem. Open-Gate system uses different data sources like object-oriented and relational, but not xml. Also, in open-gate IWRAP used is as an external interface to the middleware. MOCHA middleware system is having basic database middleware systems with components like QPC (Query Processing Coordinator), DAP (Data Access Provider). In MOCHA, QPC works as an integration server and DAP works as a translator. Here DAP is similar to wrapper or gateway, but not a pure wrapper, so scalability is not well achieved in MOCHA. As a new site is added, it must manage system-wide interactions.

As seen in literature, MOCHA and Open-Gate middleware systems are based on the general purpose database middleware systems, while other systems like SQMD, OGSA-DAI and OGSA-DQP are used for special purpose applications. For example, SQMD middleware system makes use of virtual private servers. OGSA-DQP can be used in any Grid application that aims to integrate and analyse structured data collections. In open-gate IWRAP is an external interface to the middleware. It is not the part of a middleware. So when any user wants to use open-gate, he/she must use IWRAP and is need to provide interface with IWRAP. IWRAP acts as an interface for the middleware that interacts with each data source and fetches data result. In database middleware systems, the translation of data items to the global schema is performed by wrapper or gateway. MOCHA middleware system is having basic database middleware systems with components like QPC, DAP. In MOCHA, QPC works as an integration server and DAP works as a translator. Here DAP is similar to wrapper or gateway, but not a pure wrapper. Scalability is not well achieved in MOCHA. As a new site is added, it must manage system wide interactions.

DisCo is a mediator based middleware system. Mediators accept queries and transform them into sub queries that are distributed to databases. The mediator also contains a query optimizer and run-time system. The query optimizer searches for the best way to execute a query on the run-time system. A wrapper supports the functionality of translating queries appropriate to the particular server, and reformatting answers (data) appropriate to each mediator. As the overall functionality is conducted by the mediator it overburdens it. It works as a query processor as well as integrator. Hence it affects the system performance. In DisCo scalability is achieved, but integration problem is still there in the system. SQMD middleware system gives architecture for scalable, distributed database system built on virtual private servers. A single query request from end-user is given to all the databases via middleware and agents, and the same query is executed simultaneously by all the databases. Here the problem is only single query is given to the middleware, so it require more time to process queries and overall processing cost increased. There is a need to check performance for different configurations for other resources and also memory for more efficient query processing in the virtualization environment. These are some drawbacks in existing middleware systems. There is a need to develop a middleware system which can be used as a general purpose middleware with scalability, reliability, autonomy, and also gives high performance with low communication cost.

There are research gaps which are stated here briefly as:

- An efficient integrator is needed for uniform access of data.
- There is a need for effective wrapper which will take part in integration process of the middleware, so that system performance gets improved.
- Need to achieve scalability with heterogeneous data sources.

This paper proposes the new middleware system SCM (SCalable Middleware) with components query processor, integrator and wrapper. This middleware is developed for heterogeneous distributed databases. Each database must have the wrapper. This wrapper interacts with each data source and fetches data result.

The rest of the paper is organized as follows: Section 2 describes the architecture of the proposed middleware system. Section 3 describes the application that we have implemented also it shows experiment and its related result. Conclusions and future work are given in section 4.

II. Proposed System

We propose a general middleware system, which has base as a database middleware system. This system is scalable, reliable, autonomous and highly efficient with less communication cost. Considering existing middleware systems in literature review we found some of the research gaps. To resolve all the problems, there is a need to develop a conceptual framework for middleware system. Figure 1 shows the organization of the major components in the Architecture of the SCM.

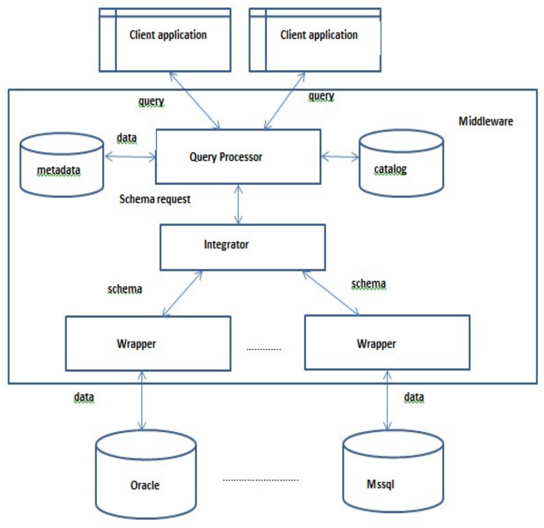


Fig 1. Conceptual framework for proposed scalable middleware system (SCM)

We have designed SCM with the major components as: Query processor (QP), Integrator and Wrapper. Here we use various data sources like oracle, mssql, mysql etc. The work of each component in the system is explained in following steps :

Step 1: Query processor (QP)

QP processes queries for client application. It requests the integrator to get the schema map. QP processes the query and sends fragments to the Ewrapper. QP controls execution of all the queries received from the client application. QP provides access to the metadata in the system. Metadata shows schema mapping for the source databases and the target databases. It keeps information of the databases, tables, columns and the users. In the catalog the log information (in terms of date, time, class, operation and response time) is recorded for each operation like insertion, deletion, updation etc. This catalog entry is used to understand the behavior and work of the various operations performed by the system. The main functions of QP are described in following steps:

- It receives query from client for processing
- It requests the integrator to get the schema map
- QP processes the query.
- QP sends fragments to the Ewrapper.

Step 2: Integrator

Integrator includes information of data distribution and schema map. The middleware system provides a uniform view and access interface for each data source. Integrator collects schema from wrapper and integrate it to form a global schema. If there is any change in the schema, it is transformed to the integrator and updates the same schema. When it receives schema request from the QP, it then sends the schema map to the QP. Here all different databases (oracle, Mysql and mssql) are converted into a common form. Client is unaware of this data distribution.

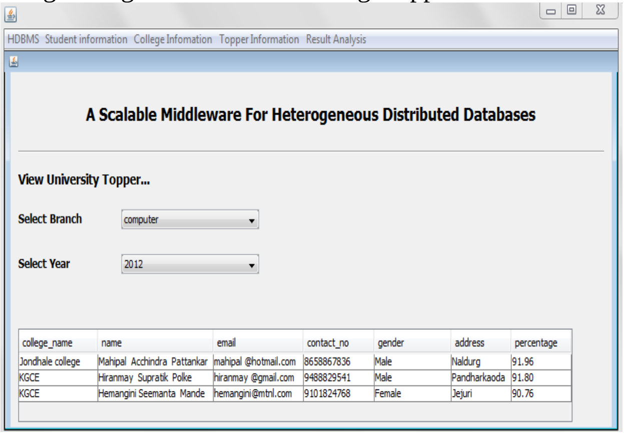
Step 3: Wrapper

Wrapper interacts with each database; gives data result. Wrapper extracts data from various databases. As per the client's requirement wrapper takes data from various databases like oracle, mysql or mssql and submits data to the integrator. Wrapper provides some classes and interfaces as DriverManager, Driver, ConnectionManager, Statement and ResultSet. DriverManager class keeps a list of database drivers. It matches connection requests from the client application with the proper database driver. Here in our system, for oracle we use oracle. jdbc. oracledriver, for mysql database driver is com. mysql. jdbc. Driver and for mssql we use jdbc. jtds. sqlserver. Driver interface handles the communication with the database server. ConnectionManager interface is used for contacting a database. The connection object is used to communicate with database. We use the Statement interface (and objects created from this interface) to submit the SQL statements to the database. Resultset object contains data retrieved from a database after executing an SQL query using Statement objects.

III. Experiments And Results

As seen in conceptual framework of proposed middleware system, we use different databases as Mysql, oracle and Mssql. We developed an application for a particular university. Here we took three colleges with three different databases for three academic years (2011, 2012, 2013) and three branches (Computer, IT, Electronics and Telecommunication) . These colleges store their data in different databases like college 1 is having mssql database, college 2 has database in Mysql, college 3 is with oracle database. The university is having the source database in Mysql. In this we get topper information and result information. Here client application is university. The client will ask query about topper and result from various colleges. If client asks for result in one of the given colleges, it will show list of students with distinction marks, first class, second class and failed students. Figure 4 shows the screen shot to view result summary. Client applications have no idea where the data located, how to connect to the data server, or what are the access interfaces. All these tasks are handled by the middleware. It provides location and access transparency. This database middleware decomposes the query and find corresponding sites for each data item which can handle the request. Schema mapping is handled by the integrator. It shows the schema mapping between the source databases and the three target databases (that is heterogeneous databases for colleges). In our application the administrator has full access to the system. The administrator can do the modifications while other users can have only the view access.

Figure 2 shows screen shot to view university topper. Here client will ask for the university topper. The result will be the first three top ranker students from the mentioned colleges. Figure 3 is to view college topper.



A Scalable Middleware For Heterogeneous Distributed Databases

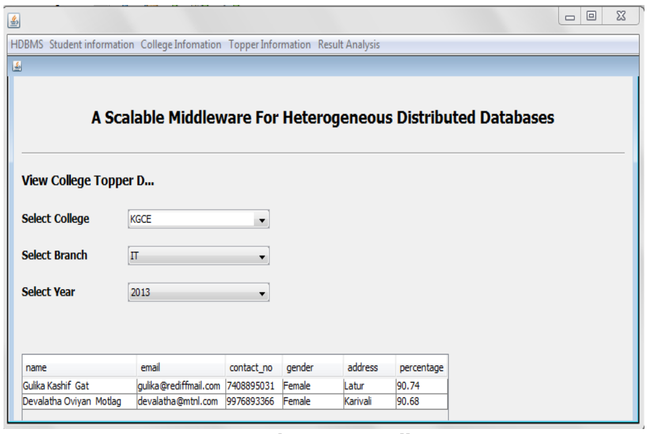
View University Topper...

Select Branch: computer

Select Year: 2012

college_name	name	email	contact_no	gender	address	percentage
Jondhale college	Mahipal Acchindra Pattankar	mahipal @hotmail.com	8658867836	Male	Naldurg	91.96
KGCE	Hiranmay Supratik Polke	hiranmay @gmail.com	9488829541	Male	Pandharkaoda	91.80
KGCE	Hemangini Seemanta Mande	hemangini@mtnl.com	9101824768	Female	Jejuri	90.76

Fig 2. Screen shot to view university topper



A Scalable Middleware For Heterogeneous Distributed Databases

View College Topper D...

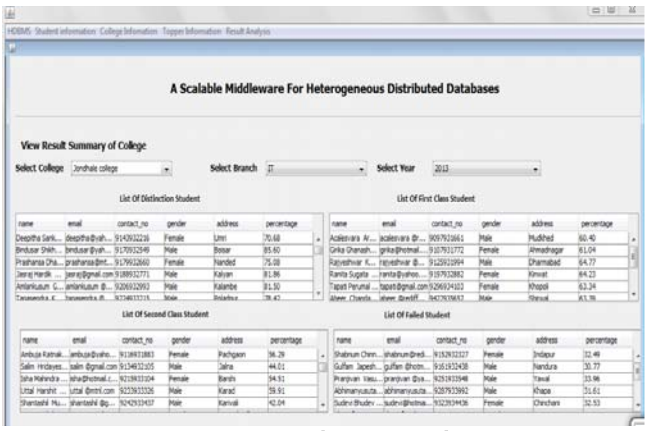
Select College: KGCE

Select Branch: IT

Select Year: 2013

name	email	contact_no	gender	address	percentage
Gulika Kashif Gat	gulika@rediffmail.com	7408895031	Female	Latur	90.74
Devalatha Oviyan Motlag	devalatha@mtnl.com	9976893366	Female	Karivali	90.68

Fig 3. Screen shot to view college topper


HDBMS Student information College Information Topper Information Result Analysis

A Scalable Middleware For Heterogeneous Distributed Databases

View Result Summary of College

Select College: Jondhale college · Select Branch: IT · Select Year: 2013

List Of Distinction Student

name	email	contact_no	gender	address	percentage
Deepitha Sark...	deepitha@yah...	9142932216	Female	Umri	70.68
Bindusar Shikh...	bindusar@yah...	9170932949	Male	Bosar	85.60
Prashansa Cha...	prashansa@mt...	9179932660	Female	Nanded	75.08
Jeeraj Hardik ...	jeeraj@gmail.com	9188932771	Male	Kalyan	81.86
Amlankusum G...	amlankusum @...	9206932993	Male	Kalambe	81.50

List Of First Class Student

name	email	contact_no	gender	address	percentage
Acalesvara Ar...	acalesvara @r...	9097921661	Male	Mudkhed	60.40
Girika Ghanash...	girika@hotmail....	9107931772	Female	Ahmadnagar	61.04
Rajveshwar K...	rajveshwar @...	9125921994	Male	Dharmabad	64.77
Ranita Sugata ...	ranita@yahoo....	9197932882	Female	Kinwat	64.23
Tapati Perumal ...	tapati@gmail.com	9296934103	Female	Khopoli	63.34

List Of Second Class Student

name	email	contact_no	gender	address	percentage
Ambuja Ratnak...	ambuja@yaho...	9118931883	Female	Pachgaon	56.29
Salim Hridayes...	salim @gmail.com	9134932105	Male	Jalna	44.01
Isha Mahindra ...	isha@hotmail.c...	9219933104	Female	Barshi	54.51
Uttal Harshit ...	uttal @mtnl.com	9233933326	Male	Karad	59.91
Shantashil Mu...	shantashil @g...	9242933437	Male	Karivali	42.04

List Of Failed Student

name	email	contact_no	gender	address	percentage
Shabnum Chinn...	shabnum@redi...	9152932327	Female	Indapur	32.49
Gulfam Japesh...	gulfam @hotm...	9161932438	Male	Nandura	30.77
Pranjivan Vasu...	pranjivan @ya...	9251933548	Male	Yaval	33.96
Abhimanyusuta...	abhimanyusuta...	9287933992	Male	Khapa	31.61
Sudevi Bhudev ...	sudevi@hotma...	9323934436	Female	Chinchani	32.53

Fig 4. Screen shot to see result

Table I: Response time (in milliseconds) for various queries

Operation	**Response time for Mysql**	**Reaponse time for Oracle**	**Response time for Mssql**
Insertion	38ms	229ms	220ms
Deletion	37ms	229ms	215ms
Data retrival to view college topper	50ms	231ms	235ms
Data retrieval to view university topper	58ms	233ms	239ms
Data retrival to see result summary	58ms	236ms	242ms

Table I shows response time for records for different operations like insertion, deletion and updation. Here we used three different databases oracle, mysql and mssql. For each of these database response time is calculated for different queries such as inserting record, deletion of record or updation in the record, also we got response time for data retrieval to find list of top ranker student from colleges and university. As we seen in the table response time for mysql database is less as compared to other databases.

IV. Conclusion And Future Work

We developed the SCM, a scalable middleware system for heterogeneous distributed databases. An attempt is made to design the middleware which facilitates the communication and coordination of application components for distributed databases. This system scales very well as the number of data sources increases. There are many factors which affects the performance of the system. Data transfer rate is different for each site. Accuracy of the system depends on the integrator. Integrator includes information of data distribution and schema map. The middleware system provides a uniform view and access interface for each data source. Response time is calculated for each record and for different queries like insertion, deletion, updation. Only the admin user can do the modification in the system. Other users have only view access, they have no rights to do any modification in the system. The work of wrapper could be improved to minimize the communication cost. In future, distributed query cost analysis can be done and it can be improved to work for the complex queries.

References

[1] S. Chavan, M. Sonawane, A. Chavan, and M. Sarjare. " Middleware: An architecture for distributed systems services".

[2] R. Elmasri and S. Navathe. Fundamentals of database systems , 2009.

[3] S. Krakowiak. "Middleware architecture with patterns and frameworks". 2007.

[4] M. Tamer O¨ zsu and P. Valduriez. Principles of distributed database systems. Springer Science+ Business Media, 2011.

[5] Kim, M. E. Pierce, and R. Guha. "Sqmd: Architecture for scalable, distributed database system built on virtual private servers". In eScience, 2008. eScience'08. IEEE Fourth International Conference on, pages 658–665. IEEE, 2008.

[6] X. Liu, Y. Shi, Y. Xu, Y. Tian, and F. Liu. " Heterogeneous database integration of epr system based on ogsa-dai. In High Performance Computing and Applications", pages 257–263. Springer, 2010.

[7] N. Reda, M. Ghaleb, and F. Fayed. "Open-gate: An efficient middleware system for heterogeneous distributed databases. International Journal of Computer Applications", 45(2), 2012.

[8] M. Rodr´ıguez-Mart´ınez and N. Roussopoulos " Mocha: a self-extensible database middleware system for distributed data sources". In ACM SIGMOD Record, volume 29, pages 213–224

[9] A. Tomasic, L. Raschid, and P. Valduriez. "Scaling heterogeneous databases and the design of disco. In Distributed Computing Systems", 1996., Proceedings of the 16th International Conference on, pages 449–457. IEEE, 1996.

[10] H. Xiang. "Integrated queries over a heterogeneously distributed scientific database using ogsa-dqp". In Information Technology and Artificial Intelligence Conference (ITAIC), 2011 6th IEEE Joint International, volume 1, pages 421–425. IEEE, 2011.