International Journal of Computer and Electrical Engineering, Vol. 2, No. 5, October, 2010
1793-8163

A Framework for Creating Global Schema Using Proxy Tables from Distributed Heterogeneous Relational Databases in Multidatabase System

Mohammad Ghulam Ali, *Member IAENG, IACSIT*

***Abstract*—Creating a proxy table, this is an alternative way of looking at the data in one or more tables from different geographical located local component heterogeneous relational databases. Proxy tables provide indirect access to the data in remote tables. Location transparency of remote data will be maintained by creating a proxy table. The proxy table will be mapped with the remote table and view. The schema for proxy table resides in the multidatabase system but data resides in the remote locations. In this paper, we propose a framework to construct a Global Schema with a set of proxy tables created from different remote local databases. We can further merge two or more proxy tables to get results from two or more remote tables located in different sites. We have also illustrated the method of creating Global Schema with an example. Query submitted on proxy table without join, union or intersect will straightway go to the respective remote table or view and will send back results to the concerned user. Query submitted using join, union or intersect on two or more proxy tables will be decomposed into a set of sub queries and will go similarly to respective remote tables or views and individual sub-query will be executed at each remote sites and finally all sub-queries will be composed and will get back results to the concerned user.**

***Index Terms*—Multidatabase, Global Schema, Relational Database, Proxy Table, View**

I. Introduction

Keeping in mind the progress in communication and database technologies (concurrency, consistency and reliability) has increased the data processing potential. Various protocols are proposed and implemented for network reliability, concurrency, atomicity, consistency, recovery and replication. The current demand is to access data from various existing databases distributed among sites in a network. All existing databases are autonomous and evolve over times. If any organization has headquarter in any country and has many branches across the globe, wants efficient and quick retrieval of information for any kind of decision supports, the proposed framework will meet this requirement in this paper. This paper has addressed nicely by implementing single global schema multidatabase system.

The multidatabase system has gaining attention of many researchers that attempts to logically integrate several different independent distributed heterogeneous DBMSs while allowing the local DBMSs to maintain complete control of their operations. It means all existing databases are autonomous and evolve over times. In multidatabase system it should be possible to address data in more than one database by a single query. On the other hand, it should be possible for different users to have different interpretation of the same data. Thus the demand on a multidatabase is to provide an interpretation of data with same or similar meaning which have different representations.

A multidatabase system is a database system that resides on top of existing component local database systems and presents a single database illustration to its users [1,2]. The Multidatabase System usually maintains a single global database schema, which is importation of all local component databases schemas and against which users issue queries and updates.

Multidatabase System maintains only global schema and the local component database system actually maintains all user data. Creating and maintaining the global schema, which requires the use of various Database Integration techniques, is a critical issue in the multidatabase system.

Variety of approaches to schema integration have been proposed e.g. [3,4,5,6,9,11,16,17]. We propose a framework to integrate multiple local component databases using proxy table technique in the multidatabase system. We are encouraged in writing this paper to the article [7] and practically experienced using proxy tables in the live project of the client-server DBMS.

Proxy tables are key to location transparency. A proxy table is a local table in the multidatabase system containing metadata which points to a remote table or view of the local component heterogeneous relational databases.

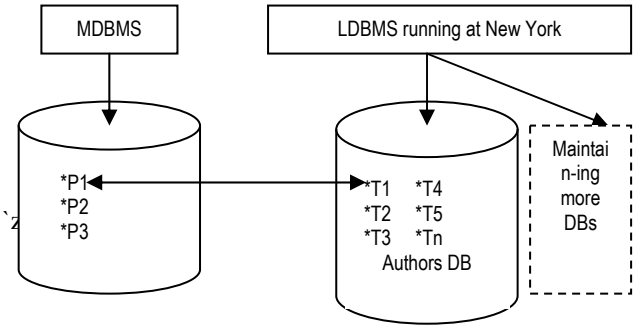
When we issue a query involving proxy tables using join i.e. where clause or union or intersect, the Multidatabase system will open connections to the remote database servers and pass the part of our query (sub-queries) involved remote tables to the remote database servers, where sub-queries run and the results sent back to the Multidatabase system. The result will be stored in the Multidatabase Server Data Cache or working storage area.

Any updates to proxy table, the update command is sent to the remote database server and the table updated there.

We illustarte examples of proxy tables and their mappings.

Mohammad Ghulam Ali, Academic Post Graduate Studies & Research, Indian Institute of Technology Kharagpur, West Bengal 721302, India (email: ali@hijli.iitkgp.ernet.in, ali_iit@yahoo.com; Phone: 91-3222-282056, 282188,220176).

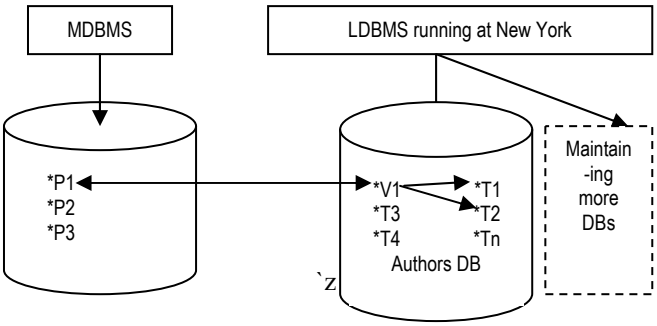
Example 1: One to one mapping between a proxy table of the Multidatabase and a table of the Local Remote Database.



Creating proxy table and establishing mapping between proxy table and remote local table.

Create proxy_table P1
external table
At "NewYork.Authors.dbo.T1"

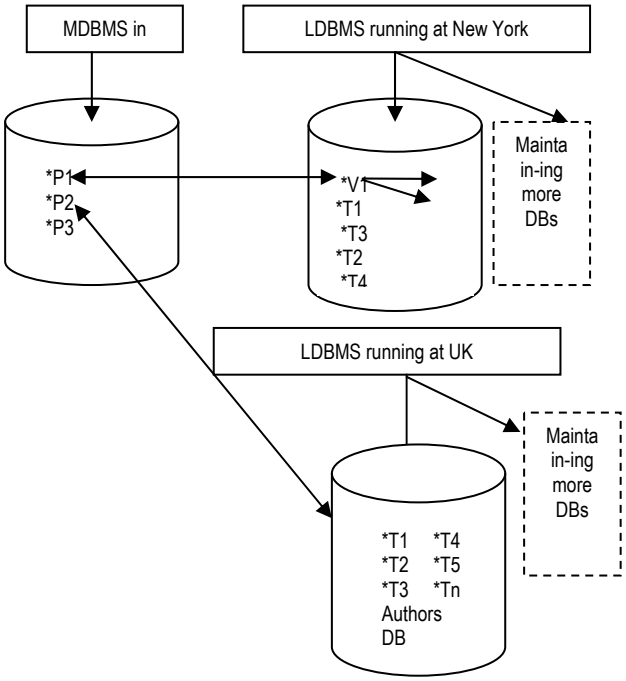
Example 2: One to one mapping between a proxy table of the Multidatabase and a view of the Local Remote Database where local view is created by two local tables.



Creating a proxy table and establishing mapping between proxy table and a remote view where a remote view is created from two remote tables of the same database server.

Create proxy_table P1
external table
At "NewYork.Authors.dbo.V1";

Example 3: Query execution using two proxy tables where 1st proxy table is directly mapped with a local remote view of the database server located at New York where remote viw is created by two tables. 2nd proxy table is directly mapped with a local remote table of a database server located at UK.



We want a list of authors with author Id and their Name from two different geographically located database servers. The following query will get back result to us:

select AuthorId, AuthorName from P1
union
select AuthorId, AuthorName from P2;

where proxy table P1 points to a view V1 at remote database server at New York in the Authors Database and a view V1 is created from two tables T1 (authors related to the computer Science) and T2 (authors related to the Mathemics) in the same database and server. Proxy table P2 points to a table T1 at UK based server in the Authors Database.

Now we propose a viable framework in this paper for the same. In our proposed framework, there is a Global Database Management System (GDBMS) where Global Schema is created with a set of proxy tables and is stored in a Global Database (GDB). A user will submit a query on Global Schema specific to any proxy table, the GDBMS will scan, parse and validate query. During this process, the system will also use Global Directory / Global Dictionary / Metadata / Global Catalog. The same query will straight forward go to the respective local remote table or view, the same query will be executed locally and at that time query will use local Directory / Dictionary / Metadata information of the local database server, and finally query will produce response and send back results to the concerned user. If user submits a join, union or intersect query on Global Schema using two or more proxy tables, the query will be decomposed into a set of sub-queries and will go to the respective remote local component database servers and each sub-query will produce a response. The sub-results coming from individual local database server are then will be composed and send back results to respective user. Similarly, through proxy table, user can update remote sites with applying SQL query on proxy table. Number of proxy table will continuously grow based on the requirements of the organization and expansion of the remote databases with

time.

II. Database Architecture Framework

The ANSI/X3/SPARC architecture [13,14,15] as shown in figure 1 is claimed to be based on the data organization in DBMS standardization. It recognizes three views of data:

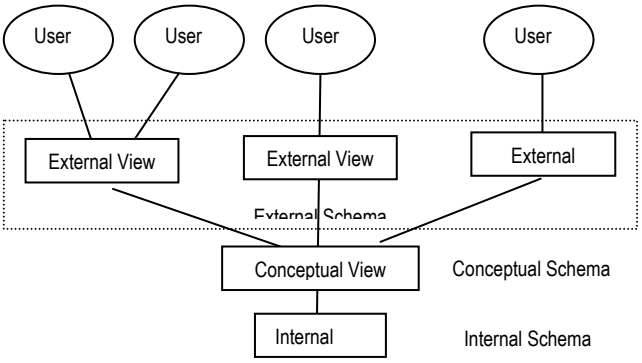


Figure 1. ANSI/SPARC Architecture

A.. Local internal schema/view

Local (internal) schema/view shows how the data is stored on each site. The format of the internal schema is dependent on the LDBMS of each site.

B. A global conceptual schema/view

A global (conceptual) schema describes the data throughout a network and shows what data is at what site. The global schema usually stored in a global directory/global dictionary/global catalog/metdata.

C. A User external schema/view

A user (external) schema/view shows how user will view and manipulate the data

III. Proposed Framwework

In our proposed framework, the Multidatabase system will control multiple gateways and will access to remote databases through these gateways. The proposed multidatabase system manages and retrieves data from multiple sites within a single graphical user interface (client-server) based application or web-based application and resides on top of the Global Schema while providing complete autonomy to individual remote database system.

The proposed framework is divided into four layers (as shown in Figure 2) based on a classical example (as shown in Figure 1) of a data-based architecture is the ANSI/SPARC model by Tsichritzis and Klug [13,14,15]. See the details work of the Multidatabase System as 4-tiered client-server model in the distributed databases [12].

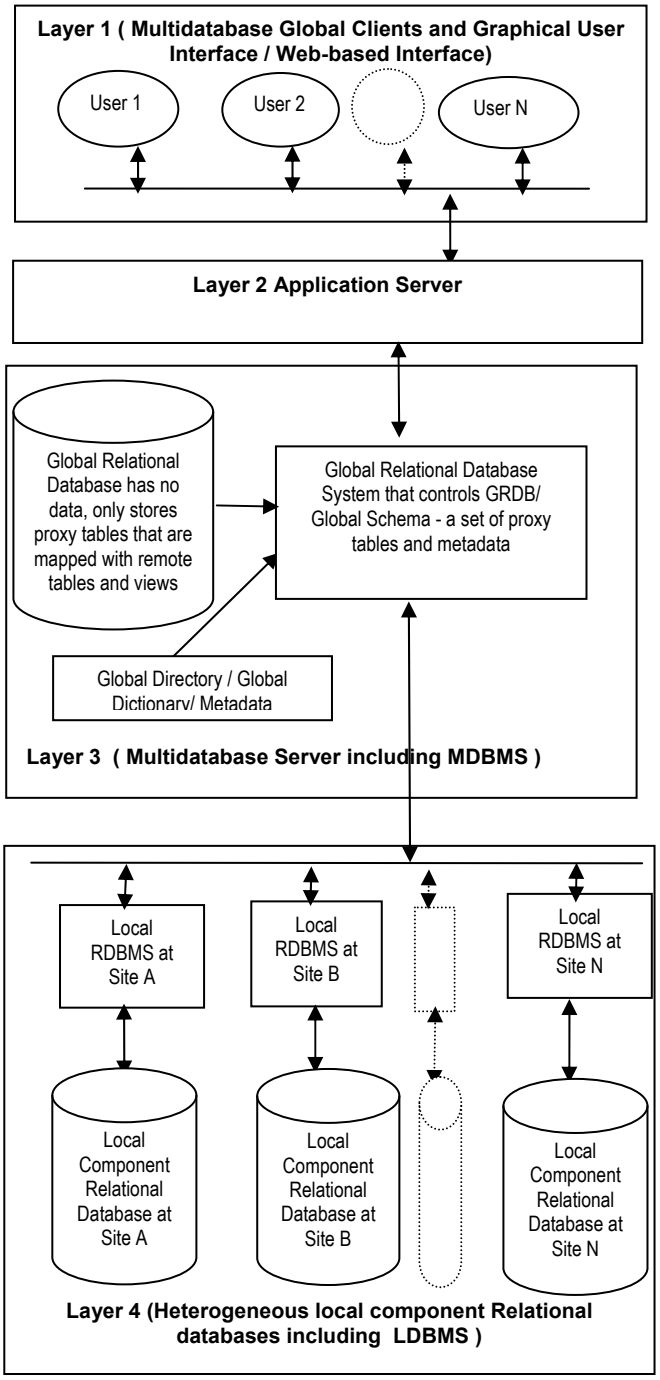


Figure 2 Proposed Framework

In our proposed framework, when proxy tables are created within the multidatabase system, metadata from the remote sites are stored within the multidatabase system tables as we named as Global Directory / Global Dictionary / Global Catalog. This metadata can be queried locally to quickly obtain information about proxy tables. This information will include column attributes, index definitions and what data objects exist at which site(s) in the distributed multidatabase system.

The maintenance of this metadata can become an issue, as the local schemas evolves over time, there should be a mechanism to import and synchronize metadata for proxy table. We will discuss about this later in this paper.

A Proxy table can also point to a view on the remote site, and it needs not come from a single table.

Through this framework, any user sitting on multidatabase system can access single ore many remote site tables located at different sites using a single query. A single query can retrieve information from many remote sites table using more than one proxy table. A single query can also update remote site table through proxy table. Any user can retrieve information from two or more remote site tables applying join, union, intersect and various way of different SQL statements query on proxy tables. Any user can transfer data from one remote site table to another remote site table through a proxy table.

Following steps are involved in the creation of Global Schema Using proxy tables from distributed heterogeneous relational databases in the multidatabase system:

A. Global Server Interface

We first define remote servers interface definition at Global Database Management System (GDMS) through which Global server will be able to interact with the remote servers. The way of server interface definition is as follows:

Database Server Interface Driver Definition

Global Server Name
Global Server IP Address
Global Server Port
Communication Protocol

Example

Global Server Name: NewYork
Global Server IP Address: 144.16.192.211
Global Server Port: 5000
Communication Protocol: TCP

B. Remote Database Gateway

Database Gateways address the needs of data access in a distributed environment. Gateways make it possible to integrate with any number of database servers. Proxy table only can be created on the basis of using SQL gateway otherwise system will not understand where to go during creation of Proxy table. The way of Remote Database Gateway definition is as follows:

Global Server Name.Database Name.Database Owner Name.Object Name

Where object's name is a name of table or view created at remote server by dbo.
Example
NewYork.HumanResourceDB.dbo.Employees

C. Method for Mapping Remote Objects to Proxy Tables and Creating Global Schema

Since actual tables and views are located at remote sites, we need to provide access methods over a network, resolving pathname i.e. remote gateway, and syntax for this, which will be transparent to the multidatabase system users. We have already discussed in A and B of III. how to define remote server interface and remote database gateway to resolve pathname of the remote site.

As we explained, earlier that global schema is a set of proxy tables. We show how to create a proxy table in the global schema and how to map proxy table to the remote objects i.e. table of view.

create proxy_table <table_name>
[external table] at "RemoteDatabaseGateway"

This will create proxy table in the multidatabase system and will derives table structure based on metadata it obtains from remote site. We use **create proxy_table** only if the table exists at the remote site, **external table** parameter defines that object is a remote table or view.
Example 1
Create proxy_table EmployeesNY
external table
At "NewYork.HumanResource.dbo.Employees"
We are creating a proxy table EmployeeNY from actual table or view which is already located at remote site in NewYork in the Human Resource Database with the name Employees.
Example 2 Alternate ways

```
create existing table EmployeesNY
(
EmployeeCode Int not null,
Name varchar(35) not null,
phone char(15) not null,
address varchar(60) null,
city varchar(25) null,
state char(2) null,
country char (3) null,
zip char (7)
)
at "NewYork.HumanResource.dbo.Employees", "table"
```

We use **create existing table** only if the table exists at the remote site. We use the same data definition language (DDL) of the remote object.
Example 3 Alternate ways

```
create table EmployeesNY
(
EmployeeCode Int not null,
Name varchar(35) not null,
phone char(15) not null,
address varchar(60) null,
city varchar(25) null,
state char(2) null,
country char (3) null,
zip char (7)
)
at "NewYork.HumanResource.dbo.Employees", "table"
```

create table command creates the table in the remote site and then creates the proxy table in the multidatabase system.

Similarly, we can create many proxy tables based on different located remote sites and in resulting of creating a Global Schema in the multidatabase system.

IV. Submit Query on Proxy Table

We can apply **select**, **insert**, **delete** and **update** SQL statements on proxy tables.

*Select * from EmployeesNY*

This will give a list of all employees from New York City located Server to the Multidatabase System.

Suppose we want list of all employees working across the globe within the same organization, then we submit a query like

*Select * from EmployeesNY*
Union
*Select * from EmployeesIND*
Union
*Select * from EmployeesLONDON*
Union
. . . . . .
*Select * from EmployeesNCountry*

Similarly we can use join query on proxy table such as

*Select * from ProxyTable1, ProxyTable2*
where
ProxyTable1.Column1 = ProxyTbale2.Column1

We can order the results query used as order by.

Similarly, we can modify data of remote site using a single SQL query on proxy table, the update command is sent to remote site and table is updated there.

We show an operator graph including query decomposition and data localization.

Case: suppose we are using more than one proxy tables and among these on proxy table is created and mapped with view of the remote site and view at remote site is again created with more than one tables using same remote site database then how query will process and will get back results to the multidatabase system.

Proxy table 1 – EmployeesNY mapped with Employees table at New York remote site

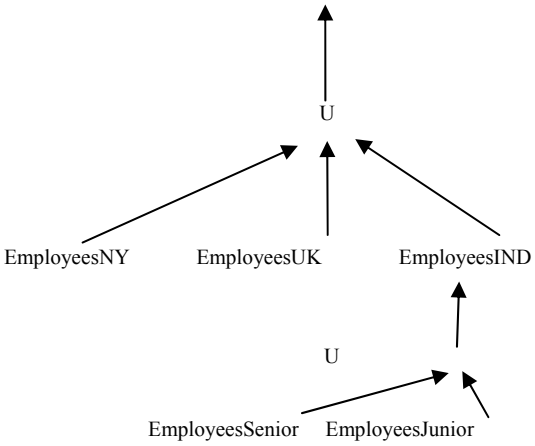
Proxy table 2 – EmployeesUK mapped with Employees table at UK remote site

Proxy table 3 – EmployeesIND mapped with Employees view at IND remote site where Employees view is created and mapped with two tables locally EmployeesSenior and EmployeesJunior.

When we submit a query as

Select EmployeeCode, Name, Country, DateOfJoining from EmployeesNY
Union
Select EmployeeCode, Name, Country, DateOfJoining from EmployeesUK
Union
Select EmployeeCode, Name, Country, DateOfJoining from EmployeesIND

The operator graph will look like as



The multidatabase system will obtain results accordingly from all remote sites. Many algorithms are proposed and implemented in distributed query processing and optimization and we are not taking into consideration this issue in this paper.

V. Database Modifications Propagation

If DBA at remote server changes the structure of the attributes such as type of the attribute, size of the attribute of the relation table will be automatically reflected in the Global Schema. Some user-defined functions may also be proposed to resolve schema conflicts. For example, attribute of one site‘s table is varchar and in other site it is char, then the user-defined function will solve this problem. I am not taking into consideration this issue in this paper. Many researchers have addressed these issues. Please see the detail work of [1,9,10,11].

VI. Database Security

The increased usage of databases to store large amounts of data has created new security problems. Typically a database contains data of various degrees of importance and levels of sensitivity. This data is shared among a wide variety of users with different responsibilities and privileges. It is therefore necessary to restrict users of the database to those portions of the total data that are necessary for their activities. Additionally, more control is needed over changes a user can make to data because of the many ways these changes can affect other users of the database [8].

The Network security expert at each remote site can better protect remote site database server by implementing a firewall between the Global Database Server and the local database server and will examine each incoming packets coming to local database server, will authenticate this and will decide whether this packet is to be denied, dropped or forwarded to local database server. Since the IP address, port and the type of network service that the Global Database Management System is using in communicating with the remote database server is known by the firewall policy rules, can easily forward, drop or deny incoming packets. The DBA at each local site will provide a better database server level and database object level security. The System Administrator at each local site will provide a better OS level security. How to exactly tackle all these issues, we do not take into consideration these issues in this paper.

VII. Conclusion

The main objective of the work is to provide transparent access to autonomous, heterogeneous and distributed local relational databases This is a viable proposed framework in the creation of Global Schema in the Multidatabase System and easy to maintain the Global Schema. In future, we plan to address other issues in the multidatabase system.

Acknowledgements

I wish to express my appreciation to reviewers of this paper who are accepting this paper for the publication in this very international journal. The have appreciated that the paper is well organized, well written, technically good and matching to journal topic.

References

[1] M. W. Bright, A. R. Hurson, and S. H. Pakzad, "A Taxonomy and Current Issues in Multidatabase Systems", *IEEE Computer*, Vol. 25, No. 3, pp. 50-60, March 1992.
[2] E. Pitoura, O. Bukhres and A. Elmagarmid, "Object-Oriented in Multidatabase Systems", *ACM Computing Surveys,* Vol. 27, No. 2, pp. 141-195, June 1995.
[3] B. Czejdo and M. Taylor, "Integration of Database System Using an Object-Oriented Approach", *Proceedings of 1st International Workshop on Interoperability in Multidatabase Systems*, April 1991, pp. 30-37.
[4] W. Gotthard, P. C. Lockemann, and A. Neufeld, "System-Guided View-Integration for Object-Oriented Databases", *IEEE Transactions on Knowledge and Data Engineering*, Vol 4, No. 1, pp. 1-22, February 1992.
[5] M. Kaul, K. Drosten, and E. J. Neuhold, "ViewSystem: Integrating Heterogeneous Information Bases by Object-Oriented Views", *Proceedings of 6th International Conference on Data Engineering*, 1990, pp. 2-10.
[6] J. L. Koh and A. L. P. Chen, "Integration of Heterogeneous Object Schemas", *in Entity-Relationship Approach, Springer-Verlag*, 1994, pp. 297-314.
[7] S. Olson, "Distributed Query Processing Using Adaptive Server Enterprise and OmniConnect 11.9.2", September, 1998.
[8] E. Bertino, and L. M. Haas, "Views and Security in Distributed Database Management Systems".
[9] M. Rehab Duwairi, "A framework for Generating and Maintaining Global Schemes in Hetrogeneous Multidatabases Systems", *IEEE*, 2003.
[10] R. Motz, and P. Fankhauser, "Propagation of semantic modifications to an integrated schema".
[11] M. G. Ali, "Object-Oriented Approach for Integration of Heterogeneous Databases in a Multidatabase System and Local Schemas Modifications Propagation", IJCSIS, Vol 6, No, 2, 2009, USA.
[12] M. G. Ali, "Multidatabase System as 4-Tiered Client-Server Distributed Heterogeneous Database System", *IJCSIS*, USA, 2009.
[13] D. Tsichritzis, and A. Klug, "The ANSI/X3/SPARC DBMS Framework Report of the Study Group on Database Management Systems. Information Systems", 1:173-191, 1978.
[14] Tamer Ozsu, Patrick Valduriez, "Principles of Distributed Database Systems".
[15] A. Umar, "Distributed Database Management Systems Issues and Approach". *The University of Michigan*, Technical Report, July 1988.
[16] Soon M. Chung, and Pyeong S. Mah, "Schema Integration for Multidatabase Using the Unified Relational and Object-Oriented Model", *ACM*, 1995.
[17] Chao Ching-Ming, "Schema Integration between Object-Oriented Databases", *Tomkang Journal of Science and Engineering*, 2001.

Mohammad G. Ali He received a degree of Master Diploma in Computer Science (1991) and Master of Science in Mathematics (1993). He has obtained 1st class in the Master of Science in Mathematics and top rank in the Computer Science in the University. He is a member of IAENG, Hong Kong and IACSIT, Singapore. He has published two papers in the International Journal of Computer Science and Information Security, USA in the month of November 2009. He has also published a paper in the Global Journal of Computer Science and Technology, USA in the month of April, 2010. His one paper is also accepted in the international conference, IASTED, ACIT-ICT 2010, Russia. He is a System Engineer Grade I in the Indian Institute of Technology Kharagpur, West Bengal, India. He is associated with System Analysis & Design, Programming, Implementation and Maintenance of Client-Server DBMSs and Web Applications Development. He is also associated with Database Administration, Web Server Administration, System Administration and Networking of the Institute. He has deployed many small to big projects in his Institute Network. His area of research is Parallel and Distributed Computing (Distributed Databases) and Software Engineering