A Comparative Study on the Performance of National Address Database (NAD) on the Grid

Mahmud Hasan[†] Md. Sadim Mahmud[††] and Yamin Mola[†††],

Dept of Computer Science & IT, Islamic University of Technology (IUT), Gazipur-1704, Bangladesh

Summary
The need for every citizen to have a valid and verifiable address and allowing diversified services to be provided to them has led to the creation of National Address Database (NAD). Traditionally, national address databases have been built and maintained at a single central location. To overcome the flaws (single point of failure, congestion and low scalability) inherent in such a centralized system our research focus is to develop a system with NAD in a grid environment. We have implemented NAD in both centralized and grid environment and done a comparative study between them. In our research we have built a test bed that describes the computational environment in which the comparison is performed. This paper includes proposed grid system architecture for NAD, tables and a graph to show the comparison. Furthermore we want to determine the database which will provide faster and more efficient response in the grid environment. For that purpose this paper also includes tables and a graph to show the comparative performance of NAD implemented in a grid environment using Intiendo Server [7] and SQL Server.

Key words:
Grid Server, Manager, Executor, Address Verification (Match Address), Web Client.

1. Introduction

As address data are produced on a local level by different private vendors, to gain access to an integrated national address database one has to buy the datasets or a subset of the datasets from the vendors. This approach can be very expensive. That is why research is being conducted to find out ways to provide address related services instead of providing the address data itself [9]. The idea of providing address related services is one of the reasons for the creation of the National Address Database (NAD). A NAD refers to a countrywide database of street addresses. Using NAD various services can be provided. Such as- Routing and vehicle navigation, spatial demographic analysis and geo-marketing, service placement and delivery, address verification and so on. Previously NAD has been built and maintained in a single central location. There are several problems associated with it such as delayed response as a result of multiple simultaneous requests, single point of failure etc.

Manuscript received July 5, 2011
Manuscript revised July 20, 2011

To solve this problem NAD can be implemented on a grid environment [5] where the database can reside at any one or all of the local authorities and will still be accessible as a national whole through the grid. This address grid will enable the creation of a virtual National Address Database. To make a valid comparison we have implemented the Address verification service (Match Address) on NAD. Besides providing the means to perform a comparative study between centralized and Grid system, the Match Address service will also serve as a proof of concept to show how various services can be provided based on NAD. The Match Address service is implemented on both centralized and Grid environment. From the response times of using this service in both centralized and our proposed system in the grid environment, we have compared the performance between them.

There is a distinct lack of service-oriented architecture-based grid computing software in this space. To overcome this limitation, we used a Windows-based grid computing framework called "Alchemi" [2][4], implemented on the Microsoft .NET [8] Platform. While the notion of grid computing is simple enough, the practical realization of grids poses a number of challenges. Key issues that need to be dealt with are heterogeneity, reliability, application composition, scheduling, resource management and security [2]. The Microsoft .NET Framework [8] provides a powerful toolset that can be leveraged for all of these, in particular support for remote execution, multithreading, security, asynchronous programming, disconnected data access, managed execution and cross-language development, making it an ideal platform for grid computing middleware. "Alchemi" was conceived with the aim of making grid construction and development of grid software as easy as possible without sacrificing flexibility, scalability, reliability and extensibility [2].

2. Proposed System Architecture in Grid Environment

We have built the system using a 3-tier architecture [1][6]. Here the tiers are defined as follows:

Tier-1: Consists of two portions. One is Grid Server or entry portal another is manager node, though it is possible to place them on different workstations.
Tier-2: Consists of several executor nodes. These PCs can be located any part of the world even users connected to the Internet. Thus we can use the unused processing time of world wide internet users. For better performance we have defined the executor architecture using Cluster Grid. Section 4 contains more discussion about this clustering technique.
Tier -3: Contains data server or database node. Here the Database is Intiendo server, developed by AfriGIS (www.afrigis.co.za). This Server reads Data from Generic Hierarchy database of National Address. We have used NAD of South Africa.

The fig.1 shows the pictorial description of this 3-tier architecture:

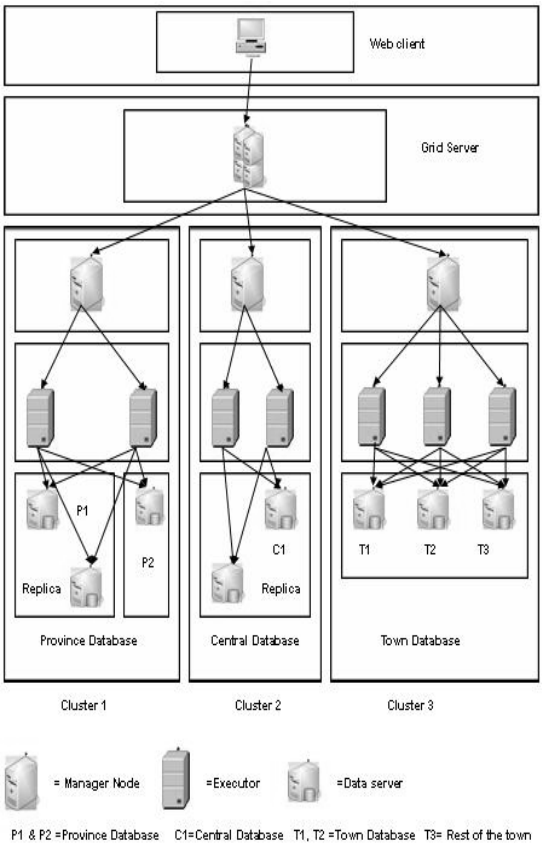

Fig.1 Proposed System Architecture

3. Components of System Architecture

Different types of nodes (or hosts) take part in desktop grid construction and application execution. Deploying a Manager node and deploying one or more Executor nodes configured to connect to the Manager construct an Alchemi desktop grid.

3.1. Manager

This node provides services associated with managing execution of grid applications and their constituent threads. The Client sends request to the Manager through a middleware (Grid Server) using a web based user interface (Web Client). This node then distributes the jobs among the working executors. Threads are scheduled on a Priority and First Come First Served (FCFS) basis [2] [3].

3.2. Executor

The Executors/Working Nodes accept threads and execute them [2]. They are actually responsible for requesting the data to the DataServer nodes and processing the data after getting them from the specified database.

3.3. Data Server Node

The DataServer node actually accesses the database. In the system we have used Intiendo server as a DataServer node which access a generic hierarchy database to retrieve data.

3.4. Web Client

Web client is a web page which provides the user interface from where the client can send the address verification request (Match Address). In Match Address service request the web page contains five fields (province, Town, Suburb, Street name, Street no.). By filling any of these fields the client can send his/her request for address verification. Upon receiving the request the system searches through the database to find a match with the field specified in the request. The system displays all the possible matched addresses with their matching percentage as a response. Web client is shown in fig-1

3.5. Grid Server

The request from the client will be directed to the grid server. We have developed this grid server as a user defined middleware that is responsible for creating threads for each of the requests sent by the client. Depending on the type of the request grid server will select the manager from a particular cluster. Then the threads created by the grid server will be sent to that manager to make

scheduling, so that each thread (user request) will be assigned to a specific executor.

4. Cluster Specification

We have introduced several clusters on this Grid. These clusters are differentiated by the request type from the end users.

4.1. Cluster 1

If a user send a request, by specifying the province name then the grid server will select manger of this cluster which in turn schedule the threads for each request to a specific executor to retrieve data from the data server node. We have assumed that there are two provinces P1 and P2 and P1 is most searched database. So we have created a replica of P1 to reduce the load. This is shown in Fig-1.

4.2. Cluster 2

If province and town name is not specified in the request then this cluster will be selected which has a central (whole database) NAD and also a replica of the database to reduce the load. This is also shown in Fig-1.

4.3. Cluster 3

If town name is specified in the request then this cluster will be selected. We have assumed that T1 and T2 are the town database which are most searched. If town name in the request matches with T1 or T2, data will be retrieved from the matched database. Otherwise the data will be retrieved from T3 which is a database for rest of the towns. Cluster 3 is shown in Fig-1.

5. System Operation

- Clients will send requests through the Web Client to the Grid Server. Grid Server will create threads for each of the request, select manager of a particular cluster depending on the request and instruct manager to generate the response. The manager will handle these requests by scheduling the threads. Each of these threads will be assigned to an executor. Executor will retrieve data from Intiendo server.
- Parameters for the request will be converted into CSV (comma separated value) format.
- Manager will send this request format to the executor.
- Grid Server will provide Manager Node with a table for the location of the DataServer node and their database content.
- The table along with the request will be provided to the executor by manger node.
- Now executor will decide which database will be required for this request and request will be sent to that data server.
- The Data server will retrieve the response from the database.
- The path through which the response will be sent to the client is as same as the request.

6. Functional Specification of the Test Bed

For the clear understanding, we are introducing the functional specification of the test bed. This specification explains the physical distribution and configuration of each node. The following nodes will form part of our NAD on the Grid. We have defined three levels for the nodes.

Level 1 Node: These types of nodes are up and available 100% of the time.

Level 2 Node: These types of nodes are and available most of the time, say 85% up-time.

Level 3 Node: These types of nodes only have dial-up access to the grid.

Node 1

Level: 1
Type: Web client
Operating System: Windows
Grid software: N/A
Street Address Data: None
Data Format: CSV (Comma Separated Value) format for Intiendo Server.
IP Address: 203.208.189.69 (Internet)

Node 2

Level: 1
Type: Grid Server (User defined middleware)
Operating System: Windows
Grid software: Alchemi
Street Address Data: None
Data Format: CSV (Comma Separated Value) format for Intiendo Server
IP Address: 172.16.24.3 (LAN)

Node 3, 4, 5

Level: 1

Type: Manager Node of cluster 1, 2 and 3
Operating System: Windows
Grid software: Alchemi
Street Address Data: None
Data Format: N/A
IP Address: 172.16.24.5 (LAN), 172.16.24.122 (LAN), 172.16.24.129 (LAN)

Node 6,7,8,9,10,11,12

Level: 1 or 2 or 3
Type: Executor Nodes of cluster 1, 2 and 3 connected to corresponding managers.
Operating System: Windows
Grid software: Alchemi
Street Address Data: None
Data Format: N/A
IP Address: 172.16.24.5 (LAN), 172.16.24.122 (LAN), 172.16.24.129 (LAN), 172.16.24.3 (LAN), 172.16.24.161 (LAN), 172.16.24.162 (LAN), 172.16.24.149 (LAN)

Node 13, 14, 15, 16, 17, 18, 19, 20

Level: 1
Type: DataServer Nodes of cluster 1, 2 and 3
Operating System: Windows
Grid software: Alchemi
Street Address Data: Street Address database of South Africa is used (Demo)
P1= Gauteng (province), P2=Western Cape (province)
C1= Demo version of the whole South Africa Street Address Data Base.
T1= Pretoria (Town), T2=Lawley (Town), T3=Rest of the town Data Base.
Data Format: Intiendo Hierarchy
IP Address: 172.16.24.5 (LAN),172.16.24.122 (LAN),172.16.24.129 (LAN), 172.16.24.3 (LAN),172.16.24.161 (LAN),172.16.24.162 (LAN),172.16.24.149 (LAN), 172.16.24.150 (LAN)

7. Results and Performance Evaluation

The test bed for grid environment is prepared and to measure the performance for Match Address service we have created specific set of requests and sent those requests simultaneously to both centralized system and our proposed system in grid environment.

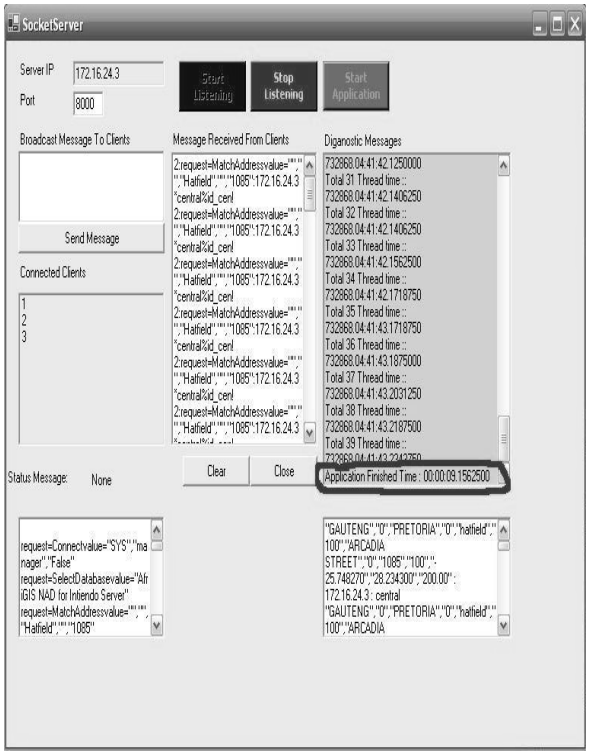

Fig.2 Snapshot of Grid Server during execution on Cluster Grid with 40 requests.

For our experiment we have sent 20 different requests to both centralized and grid system and documented the response time. This procedure has done three times and from these data average response time is calculated. Analogously for 30 and 40 requests the same procedure is followed. The snapshot of output is given in fig.2 and fig.3. Table-1 and Table-2 contains the response times for different requests in grid and centralized system respectively.

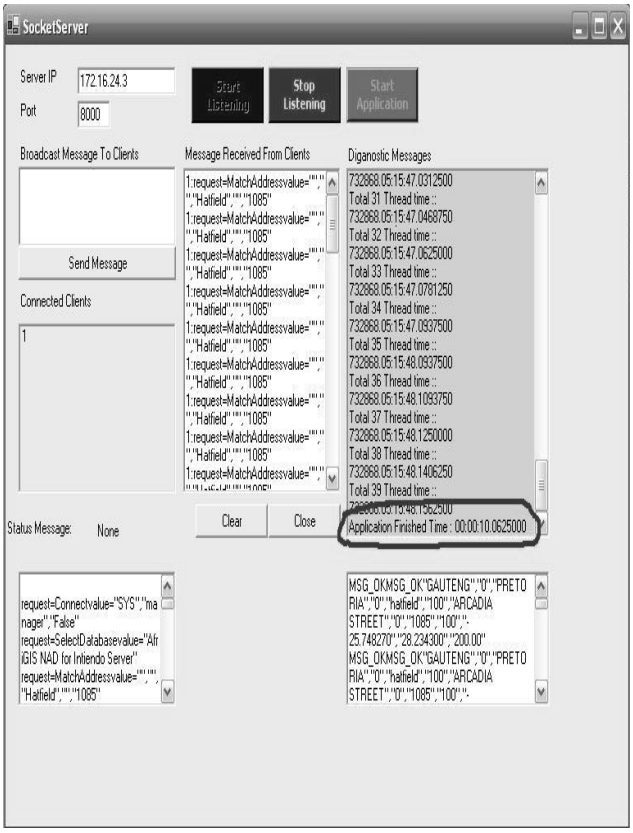

Fig.3 Snapshot of Grid Server during execution on Centralized System with 40 requests.

Table 1. Response Time Vs Number of Requests (Grid)

No of Request (REQ)	*Response Time in Cluster Grid Environment*			
	Test1	*Test2*	*Test3*	*Avg*
20	5.6093	5.3437	5.3593	5.4375
30	5.8281	5.9001	5.7561	5.8281
40	9.1562	9.1562	9.1561	9.1562

Table 2. Response Time Vs Number of Requests (Centralized)

No of Request (REQ)	*Response Time in centralized Environment*			
	Test1	*Test2*	*Test3*	*Avg*
20	5.6562	6.6718	5.6406	5.9895
30	6.75	6.59	6.43	6.59
40	10.071	10.062	10.053	10.062

By plotting average response time and no of requests from Table-1 and Table-2, a graph (fig.4) is constructed to visualize the performance.

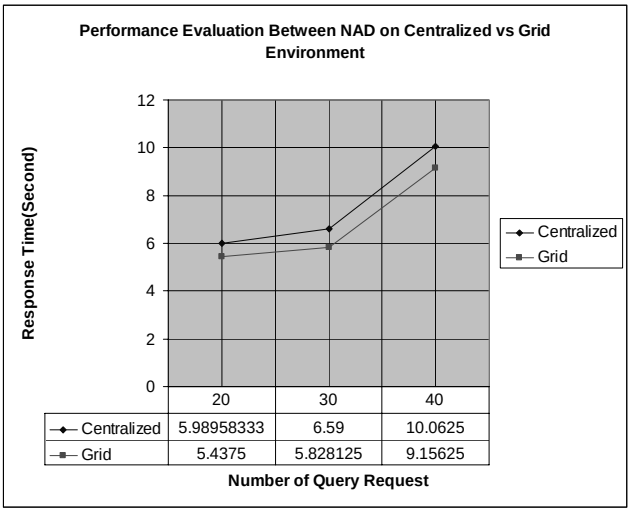

Fig.4 Performance Graph (Centralized vs. Grid)

From Table.1, Table.2 and Fig.4, it is clearly understandable that our NAD on the Grid can provide much better performance than any available NAD systems in any country. The curve for Cluster Grid is always better than the curve for centralized NAD implementation. Now for the next phase of our research similar operation has been performed to determine the faster and more efficient response providing database in grid environment. For simplicity the comparison is made between Intiendo Server and SQL Server. It can be easily concluded from Table.3, Table.4 and Fig.5 that implementation of NAD in a grid environment using Intiendo Server has considerably faster response time than the implementation of NAD in a grid environment using SQL Server.

Table 3. Response Time Vs Number of Requests (Intiendo)

No of Request (REQ)	*Response Time in Grid Environment For Intiendo Server*			
	Test 1	*Test2*	*Test3*	*Avg*
1	1.008	1.200	1.119	1.109
2	1.448	1.421	1.442	1.437
3	2.499	2.481	2.52	2.50

Table 4. Response Time Vs Number of Requests (SQL)

No of Request (REQ)	*Response Time in Grid Environment For SQL Server*			
	Test 1	*Test2*	*Test3*	*Avg*
1	2.210	2.531	2.522	2.421
2	2.980	2.864	2.991	2.945
3	4.515	4.622	4.573	4.57

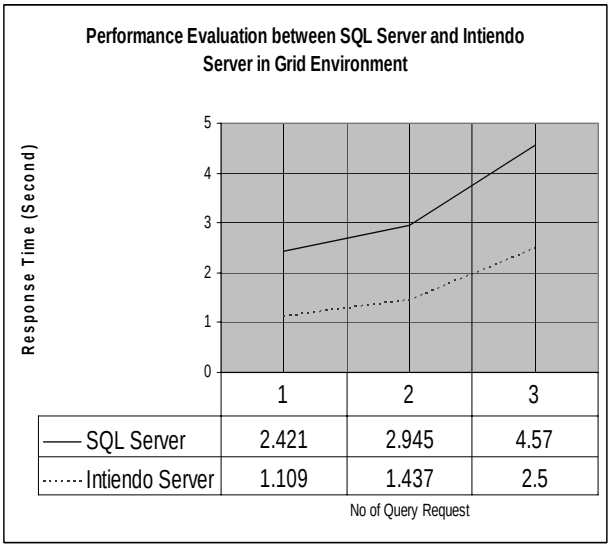

Fig.5 Performance Graph (SQL Server vs. Intiendo Server)

8. Conclusion

While this paper focuses on the performance study between NAD on a Grid and centralized environment, it also provides a service oriented cluster based architecture where various services can be implemented and integrated in order to satisfy various client requests in a Grid environment. We can conclude from our experimentation result that the performance of the grid system is better than the centralized system and grid system that uses Intiendo server as data node is more efficient than grid system that uses SQL server as data node. Though many countries are currently using NAD in various ways, they can ensure better performance if they use this architecture with Intiendo server. Although we have performed the experiments with a small number of requests, from these experiments we can infer that for a larger number of requests the reduction of the response time in our proposed grid system architecture will be significantly faster.

Acknowledgements

We used Intiendo Server, developed by AfriGIS (www.afrigis.co.za) on the database node.

References

[1] Mahmud Hasan, Yamin Mowla, Md. Sadim Mahmud and Md. Ahsan Arefin (2007) "National Address Database on the Grid", 10th International Conference on Computer and Information Technology (ICCIT) 2007, Bangladesh.

[2] Akshay Luther, Rajkumar Buyya, Rajiv Ranjan, and Srikumar Venugopal (2004) Alchemi: A .NET-based Grid Computing Framework and its Integration into Global Grids.

[3] Ian Foster, Carl Kesselman, and S. Tuecke, (2001)The Anatomy of the Grid: Enabling Scalable Virtual Organizations, International Journal of Supercomputer Applications, 15(3), Sage Publications, 2001, USA.

[4] Ahsan Arefin, Shiblee Sadik, Judith Bishop, Serena Coetzee, (2006) "Alchemi Vs Globus: The Performance Comparison", 4th International Conference of Electrical and Computer Engineers (ICECE) 2006, Bangladesh.

[5] Md. Ahsan Arefin, Md. Shiblee Sadik (2006), "Implementation of Server on GRID: A Supercomputer Approach", 15th International Conference of Information Systems and Developmet (ISD) 2006, Budapest, Hungary.

[6] Md. Ahsan Arefin, Md. Shiblee Sadik, Md. Forhad Rabbi, M. A. Mottalib (2007), "3-Tier Architecture of Data Server on Grid: Implemented Using Globus Toolkit", International WORLDCOMP Conference of Grid Computing Application (GCA, 07), Las Vegas, USA.

[7] AfriGIS - Home, www.afrigis.co.za. Accessed: 10 April, 2011.

[8] Microsoft dot NET framework, www.microsoft.com, Accessed: 15 March, 2011.

[9] Serena Coetzee, Judith Bishop (2009), "Address databases for national SDI: Comparing the novel data grid approach to data harvesting and federated databases", International Journal of Geographical Information Science 23(9): 1179-1209 (2009).

Mahmud Hasan received BSc in Computer Science and Information Technology degree from Islamic University of Technology (IUT) in 2007. He has completed his MSc degree in an Erasmundus Mundus program titled "Network and E-Business Centred Comuting (NeBcc)" in 2011. Currently he is a Lecturer of the Department of Computer Science and IT (CIT) at Islamic University of Technology (IUT).

Md. Sadim Mahmud received BSc in Computer Science and Information Technology degree from Islamic University of Technology (IUT) in 2007. Currently he is pursuing his MSc degree in the department of Computer Science at the University of Western Ontario.

Yamin Mowla received BSc in Computer Science and Information Technology degree from Islamic University of Technology (IUT) in 2007. Currently he is working as an associate engineer in NSS Operation and Maintenance department of a cellular service provider in Bangladesh known as Banglalink.