Framework for Client-Server Distributed Database System for Licensing and Registration of Automobiles in Nigeria.

Olabode Olatubosun, Ph.D.[1*] and Solanke Olakunle, M.Tech.[2]

[1]Computer Science Department, Federal University of Technology, Akure, Nigeria, GSM: 08033511257.
[2]Department of Mathematical Sciences, Olabisi Onabanjo University, Ago-Iwoye, Nigeria.

[*]E-mail: olabode_olatubosun@yahoo.co.uk

ABSTRACT

Registration is the process of adding a vehicle to the Motor Vehicle Register and issuing it registration plates. Vehicle licensing is the payment of a fee for the use of a motor vehicle on public roads. In recent times, it has been observed that a reduction in Nigerian government internally generated revenue is in part a result of the evasion of automobile vehicle property and usage taxes by individuals who abuse or circumvent the state's automobile registration system. In this research work, an attempt is made to design a framework for a client server distributed database system for licensing and registration of automobiles in Nigeria.

The system consists of a relational database of automobile decision variables which could be shared by the three level organizations, viz, Vehicle Inspection Office (VIO), Federal Road Safety Commission, and the Board of Internal Revenue. Each of these organizations is considered as a site. The Database is expected to be hosted by the VIO while the concept of data replication and fragmentation is adopted by other sites to have access to data/records in the database. The system is intelligent and capable of checking to detect multiple registrations, registration of stolen automobiles, malicious registrations, registration of damaged or reformed automobiles, and fictitious registration. The system is capable of generating reports for decision makers to enable monitoring and enforcement.

(Keywords: distributed database system, licensing, automobile, fragmentation, replication, registration)

INTRODUCTION

Navathe (2000) described a distributed database (DDB) as a collection of multiple logically interrelated databases distributed over a computer network, and a distributed database management system (DDBMS) as a software system that manages a distributed database while making the distribution transparent to the user. From the definition of Navathe (2000), the elementary unit of a distributed system is a computer that is networked with other computers; the computer being autonomous in the way it carries out its actions.

Computers are linked to one another over a communications network that enables an exchange of messages between computers. The objective of this message exchange is to achieve cooperation between computers for the purpose of attaining a common goal. In this research, an attempt is made to design a framework for a client server distributed database system for licensing and registration of automobiles in Nigeria.

The system consists of a relational database of automobile decision variables which could be shared by the three level organizations, viz, Vehicle Inspection Office (VIO), Federal Road Safety Commission, and the Board of Internal Revenue. Each of these organizations is considered as a site. The Database is expected to be hosted by the VIO while the concept of data replication and fragmentation is adopted by other sites to have access to data/records in the database. The system is intelligent and capable of checking to detect multiple registrations, registration of stolen automobile, malicious registration, and registration of damaged or reformed automobile and fictitious registration. The system is capable of generate reports for decision makers to enable monitoring.

The client-server architecture is used as a platform for database application development and is an approach which has successfully been used to solve some lingering problems in society

today. The Client Server system is adopted in this research paper.

THE AUTOMOBILE LICENSING AND REGISTRATION SYSTEM IN NIGERIA

Registration is the process of adding a vehicle to the Motor Vehicle Register and issuing it with registration plates. Vehicles must be registered in order to be used on public roads and only require re-registration if the registration has been cancelled. Vehicle licensing is the payment of a fee for the use of a motor vehicle on public roads. When the fee is paid, you receive a label indicating the expiry date of the license. This license label must be displayed on the vehicle.

In Nigeria, three arms of government are responsible for automobile licensing, registration and control. These include the state board of internal revenue, the local government office and the police force. The usual practice is for an owner to visit these three arms for the necessary payment, data collection, and issuance of necessary documents and materials such as plate numbers. The state board of internal revenue collects taxes for new automobile licenses and registrations from owners through a designated bank. They may request some documents such as custom papers, purchase receipt, or a change of owner certificate on automobile.

Every automobile within the nation must be registered under a state and a local government before a license plate is issued. Nigerian automobile registration plates often have the state written at the top, and have a group of three letters at the right, indicating the district of registration (followed by their main town to aid location, for example AH 35 KTP). License plates serve to help law enforcement, motor vehicle authorities, and others identify a vehicle, while simultaneously indicating that the registrant has paid the proper registration fee and taxes on the automobile.

License plates also offer information such as the weight class, the county, state and local government in which the vehicle is registered, use restrictions (private or commercial), and the age and weight of the vehicle. In addition, some license plates show whether the owner of the vehicle is a member of a special organization or group such as the police force and the Federal Road Safety Commission. Moreover proof of ownership certificates are issued to owners of automobiles on payment of certain fee by the board. Automobiles in this context include personal vehicles (car, trucks, and trailers) and motorcycles. With this traditional approach, several million Naira per year may be lost to Nigerian state and local governments as a result of the evasion of automobile vehicle property and usage taxes by individuals who abuse or circumvent the state's automobile registration system. These evaders may include highly influential people in the society, government, uniformed personnel, and commercial drivers.

Police registration involves physical inspection of the automobile to obtain some important information about the automobile. The obtained information is then recorded by the police while a certificate of registration or police clearance (ECMR) is issued to the owner of the automobile.

THEORY OF DISTRIBUTED DATABASE AND DDBMS

A *distributed system* is an information-processing system that contains a number of independent computers that cooperate with one another over a communications network in order to achieve a specific objective (Kay-Romer et al., 2006). Distributed databases bring the advantages of distributed computing to the database management domain. A distributed computing system consists of a number of processing elements, not necessarily homogeneous, that are interconnected by a computer network, and that cooperate in performing certain assigned tasks.

A physical view of a distributed system includes computers as nodes of the communications network along with details about the communications network itself. In contrast, a logical view of a distributed system highlights the applications aspects.

Figure 1 can therefore also be interpreted as a set of cooperating processes. The distribution aspect refers to the distribution of state (data) and behavior (code) of an application. The process encapsulates part of the state and part of the behavior of an application, and the application's semantics are achieved through the cooperation of several processes. The logical distribution is independent of the physical one.

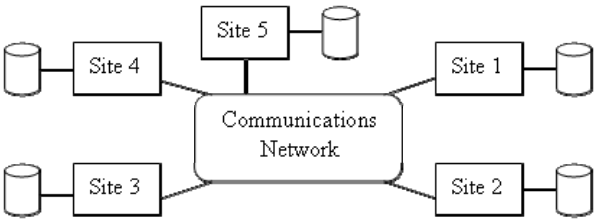

Figure 1: A Distributed Database Architecture.

Distributed database management has been proposed for various reasons ranging from organizational decentralization and economical processing to greater autonomy. Navathe (2000) highlighted some of these advantages as presented below.

a. *Management of distributed data with different levels of transparency such as*

   - *Distribution or network transparency:* This refers to freedom for the user from the operational details of the network. It may be divided into location transparency and naming transparency. Location transparency refers to the fact that the command used to perform a task is independent of the location of data and the location of the system where the command was issued. Naming transparency implies that once a name is specified, the named objects can be accessed unambiguously without additional specification.

   - *Replication transparency:* In this case, copies of data may be stored at multiple sites for better availability, performance, and reliability. Replication transparency makes the user unaware of the existence of copies.

   - *Fragmentation transparency:* Two types of fragmentation are possible. Horizontal fragmentation distributes a relation into sets of tuples (rows). Vertical fragmentation distributes a relation into sub-relations where each sub-relation is defined by a subset of the columns of the original relation. A global query by the user must be transformed into several fragment queries. Fragmentation transparency makes the user unaware of the existence of fragments.

b. *Increased reliability and availability:* These are two of the most common potential advantages cited for distributed databases. Reliability is broadly defined as the probability that a system is running (not down) at a certain time point, whereas availability is the probability that the system is continuously available during a time interval. When the data and DBMS software are distributed over several sites, one site may fail while other site continues to operate. Only the data and software that exists at the failed site cannot be accessed. This improves both reliability and availability. Further improvement is achieved by judiciously *replicating* data and software at more than one site. In a centralized system, failure at a single site makes the whole system unavailable to all users. In a distributed database, some of the data may be unreachable, but users may still be able to access other parts of the database.

c. *Improved performance:* A distributed DBMS fragments the database by keeping the data closer to where it is most often needed. Data localization reduces the contention for CPU and I/O services and simultaneously reduces access delays involved in wide area networks. When a large database is distributed over multiple sites, smaller databases exist at each site. As a result, local queries and transactions accessing data at a single site have better performance because of the smaller local databases. In addition, each site has a smaller number of transactions executing than if all transactions are submitted to a single centralized database. Moreover, inter-query and intra-query parallelism can be achieved by executing multiple queries at different sites, or by breaking up a query into a number of sub-queries that execute in parallel. This contributes to improved performance.

d. *Easier expansion:* In a distributed environment, expansion of the system in terms of adding more data, increasing database sizes, or adding more processors is much easier.

In Navathe (2000), the term distributed database management system can describe various systems that differ from one another in many

respects. The main thing that all such systems have in common is the fact that data and software are distributed over multiple sites connected by some form of communication network. Different types of DDBMSs and the criteria and factors that make some of these systems different have been discussed in the literature.

The first factor considered is the degree of homogeneity of the DDBMS software. If all servers (or individual local DBMSs) use identical software and all users (clients) use identical software, the DDBMS is called homogeneous; otherwise, it is called heterogeneous. Another factor related to the degree of homogeneity is the degree of local autonomy. If there is no provision for the local site to function as a stand-alone DBMS, then the system has no local autonomy. On the other hand, if *direct access* by local transactions to a server is permitted, the system has some degree of local autonomy.

At one extreme of the autonomy spectrum, we have a DDBMS that "looks like" a centralized DBMS to the user. A single conceptual schema exists, and all access to the system is obtained through a site that is part of the DDBMS—which means that no local autonomy exists. At the other extreme we encounter a type of DDBMS called a *federated DDBMS* (or a *multi-database system*). In such a system, each server is an independent and autonomous centralized DBMS that has its own local users, local transactions, and DBA and hence has a very high degree of *local autonomy.*

The term federated database system (FDBS) is used when there is some global view or schema of the federation of databases that is shared by the applications. On the other hand, a multi-database system does not have a global schema and interactively constructs one as needed by the application. Both systems are hybrids between distributed and centralized systems and the distinction we made between them is not strictly followed. We will refer to them as FDBSs in a generic sense.

In a heterogeneous FDBS, one server may be a relational DBMS, another a network DBMS, and a third an object or hierarchical DBMS; in such a case it is necessary to have a canonical system language and to include language translators to translate sub-queries from the canonical language to the language of each server. The type of heterogeneity present in FDBSs may arise from several sources, viz; differences in data models, differences in constraints and differences in query languages. Semantic heterogeneity occurs when there are differences in the meaning, interpretation, and intended use of the same or related data. Semantic heterogeneity among component database systems (DBSs) creates the biggest hurdle in designing global schemas of heterogeneous databases.

There are three alternative approaches to separating functionality across different DBMS-related processes; these alternative distributed DBMS architectures are called Client-Server, Collaborating Server, and Middleware.

A **Client-Server** system has one or more client processes and one or more server processes, and a client process can send a query to any one of the server process. Clients are responsible for user-interface issues, and servers manage data and execute transactions. Thus, a client process could run on a personal computer and send queries to a server running on a mainframe.

In a **Collaborating Server** system, we can have a collection of database servers, each capable of running transactions against local data, which cooperatively execute transactions spanning multiple servers. When a server receives a query that requires access to data at other servers, it generates appropriate sub-queries to be executed by other servers and puts the results together to compute answers to the original query. Ideally, the decomposition of the query should be done using cost-based optimization, taking into account the costs of network communication as well as local processing costs.

The Middleware architecture is designed to allow a single query to span multiple servers, without requiring all database servers to be capable of managing such multi-site execution strategies. It is especially attractive when trying to integrate several legacy systems, whose basic capabilities cannot be extended. The idea is that we need just one database server that is capable of managing queries and transactions spanning multiple servers; the remaining servers only need to handle local queries and transactions. We can think of this special server as a layer of software that coordinates the execution of queries and transactions across one or more independent database servers; such software is often called **middleware**. The middleware layer is capable of executing joins and other relational operations on

data obtained from the other servers, but typically, does not itself maintain any data.

FRAMEWORK FOR THE CLIENT-SERVER ARCHITECTURE OF AUTOMOBILE REGISTRATION AND LICENSING DISTRIBUTED DATABASES

The primary types of system architectures for information processing include: Service Oriented Architecture (SOA), distributive (client–server), and centralized information systems processing more commonly associated with mainframe and midrange computers. The Client–Server Architecture is considered in this case. This is a network architecture in which each computer or process on the network is either a client or a server. Servers are powerful computers or processes dedicated to managing disk drives (file servers), printers (print servers), or network traffic (network servers). Clients are PCs or workstations on which users run applications. Clients rely on servers for resources, such as files, devices, and even processing power.

Interaction between client and server might proceed as follows during the processing of an SQL query:

a. The client parses a user query and decomposes it into a number of independent site queries. Each site query is sent to the appropriate server site.

b. Each server processes the local query and sends the resulting relation to the client site.

c. The client site combines the results of the sub-queries to produce the result of the originally submitted query.

In this approach, the SQL server has also been called a transaction server (or a database processor (DP) or a back-end machine), whereas the client has been called an application processor (AP) (or a front-end machine). The interaction between client and server can be specified by the user at the client level or via a specialized DBMS client module that is part of the DBMS package. For example, the user may know what data is stored in each server, break down a query request into site sub-queries manually, and submit individual sub-queries to the various sites.

The resulting tables may be combined explicitly by a further user query at the client level. The alternative is to have the client module undertake these actions automatically.

In a typical DDBMS, it is customary to divide the software modules into three levels:

a. The server software is responsible for local data management at a site, much like centralized DBMS software.

b. The client software is responsible for most of the distribution functions; it accesses data distribution information from the DDBMS catalog and processes all requests that require access to more than one site. It also handles all user interfaces.

c. The communications software (sometimes in conjunction with a distributed operating system) provides the communication primitives that are used by the client to transmit commands and data among the various sites as needed. This is not strictly part of the DDBMS, but it provides essential communication primitives and services.

The client is responsible for generating a distributed execution plan for a multi-site query or transaction and for supervising distributed execution by sending commands to servers. These commands include local queries and transactions to be executed, as well as commands to transmit data to other clients or servers. Hence, client software should be included at any site where multi-site queries are submitted. Another function controlled by the client (or coordinator) is that of ensuring consistency of replicated copies of a data item by employing distributed (or global) concurrency control techniques. The client must also ensure the atomicity of global transactions by performing global recovery when certain sites fail.

SYSTEM DESIGN

A list of some documents and information requested at the various stages of the design are presented below:

a. Registration Identity of the automobile
b. Application Code
c. Passport

d. State where the automobile is been registered
e. Local government area where the automobile is been registered
f. Automobile Make
g. Automobile Model
h. Automobile Type
i. Color of automobile
j. Chassis number
k. Engine number
l. Number of Cylinders of the automobile
m. Engine Capacity
n. Purpose for which the automobile is been registered
o. Inspection date
p. Road certificate number
q. Testing authority
r. Previous Registration number
s. License Office
t. Owner Name
u. Owner Status
v. Owner Address
w. Other related information.

A careful study and analysis of the above data suggests that while the three governmental arms collect data differently and through disjointed efforts, they all basically require the same set of data. Data collection is quite laborious since record keeping is associated with manual data processing. In this research, an attempt was made to formulate a standard operational procedure for the three sectors involved in automobile registration so as to bring effectiveness and reduce duplication of information involved in managing automobile registrations. Moreover, a centrally managed database and a neural network solution are presented with an element of multi-level data access.

APPLICATION PROCEDURE

In this framework, the flow of registration includes, the owner presenting the automobile for inspection at the VIO. Forms are issued to the owner to extract the owner's and the vehicle's details. One reason vehicles need to be registered is to ensure that only those meeting safety standards are on the road. When a vehicle is registered it undergoes safety and identification inspections. Thereafter, a certificate of roadworthiness is issued for the vehicle.

Registration also allows information about the vehicles on our roads to be recorded on the Motor Vehicle Register. The information helps enforcement and deters vehicle theft. Sometimes, if a vehicle's registration has been cancelled and you want to use the vehicle on the road again, it must be re-registered. Common reasons for cancelling registration include vehicles being 'written off' by insurance companies, destroyed, or rendered permanently useless or permanently removed from Nigerian roads. The re-registration process involves a safety inspection, certification, registration, and licensing. The owner needs to provide proof that the vehicle has previously been registered.

The owner proceeds to make online payment from the designated agent. On payment, a certificate of payment (detail of payment) is sent electronically to the vehicles owner record in the database with the VIO, is issued to the payer. The owner then proceeds to the Federal Road Safety Commission to collect the necessary vehicle licensing documents and the plate number. Figure 2 presents the physical flow of the systems functionality.

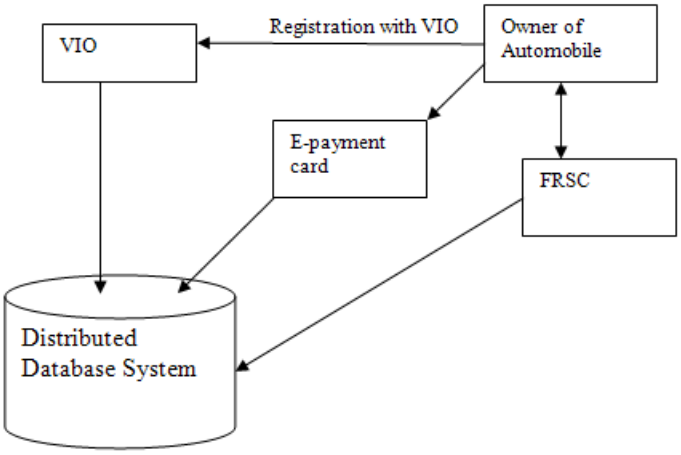

Figure 2: Data Flow of the Distributed Database System.

In Date, (1986), a **relation** (or **relation state**) r of the relation schema $R(A_1, A_2, . . ., A_n)$, also denoted by r(R), is a set of n-tuples $r = \{t_1, t_2, . . ., t_m\}$. Each **n-tuple** t is an ordered list of n values $t = <v_1, v_2, . . ., v_n>$, where each value v_i, 1 1 i 1 n, is an element of dom(A_i) or is a special **null** value. The i^{th} value in tuple t, which corresponds to the attribute A_i, is referred to as t[A_i]. One of

the relations in this database is as presented below:

FRSC (RegistrationId, ApplicationCode, Passport, State, Lga, VehicleMake, VehicleModel, VehicleType, Color, ChasisNo, EngineNo, CylinderNo, EngineCapacity, VehiclePurpose, InspectionDate, RoadCertificateNo, TestingAuthority, PreveousRegNo, LicenseOffice, OwnerFName, OwnerInit OwnerLName OwnerStatus, OwnerAddress, OwnerTown, OwnerPhone, ApplicantName, ApplicantAddress, ApplicationDate, ApplicantEmail, ApplicantId, NumberType, NumberAlloted, Payment, PaymentStatus, PaymentRefCode, PaymentDate, AuthorisingOfficer, CodeNo, Checked)

COMPONENTS OF THE DISTRIBUTED SYSTEM

The distributed system has Internet facing Web-based applications that can be accessed remotely by the users either within the confines of the organization or remotely. The following is list of the information technology (IT) infrastructure components of the system

Firewall: A system designed to prevent unauthorized access to or from a private network. Firewalls can be implemented in both hardware and software, or a combination of both. Firewalls are frequently used to prevent unauthorized Internet users from accessing private networks connected to the Internet, especially intranets. All messages entering or leaving the intranet pass through the firewall, which examines each message and blocks those that do not meet the specified security criteria. There are several types of firewall techniques:

- *Packet Filter*: Looks at each packet entering or leaving the network and accepts or rejects it based on user-defined rules. Packet filtering is fairly effective and transparent to users, but it is difficult to configure. In addition, it is susceptible to Internet Protocol (IP) spoofing.

- *Application Gateway*: Applies security mechanisms to specific applications, such as File Transfer Protocol (FTP) and Telnet servers. This is very effective but can impose performance degradation.

- *Circuit-Level Gateway*: Applies security mechanisms when a Transmission Control Protocol (TCP) or User Datagram Protocol (UDP) connection is established. Once the connection has been made, packets can flow between the hosts without further checking.

- *Proxy Server*: Intercepts all messages entering and leaving the network. The proxy server effectively hides the true network addresses.

Router: A router is a special purpose computer or software device that enables two or more dissimilar networks to communicate. Routers route traffic, which consists of Transmission Control Protocol/Internet Protocol (TCP/IP) packets.

Host: A computer that is connected to a TCP/IP network, including the Internet.

Server(s): A server is a dedicated computer that allows other computers to connect to it. Various types of servers exist which include the following:

- Domain Name System
- Web servers
- Internet banking servers
- E-mail servers
- Proxy servers

PC Workstations: In networking, a workstation refers to any computer connected to a local area network. It could be a workstation or a personal computer.

Intrusion Detection Systems: Intrusion detection is fundamentally the process of monitoring computer networks and systems for violations of computer policy.

NETWORK STRUCTURE

A metropolitan area network (MAN), which can cover a city, is best for this kind of system. This system grew from earlier community antenna systems used in areas with poor over-the-air television reception. In these early systems, a large antenna was placed on top of a nearby hill and the signal was then piped to the subscribers' homes. Figure 3 describes the structure of this framework.

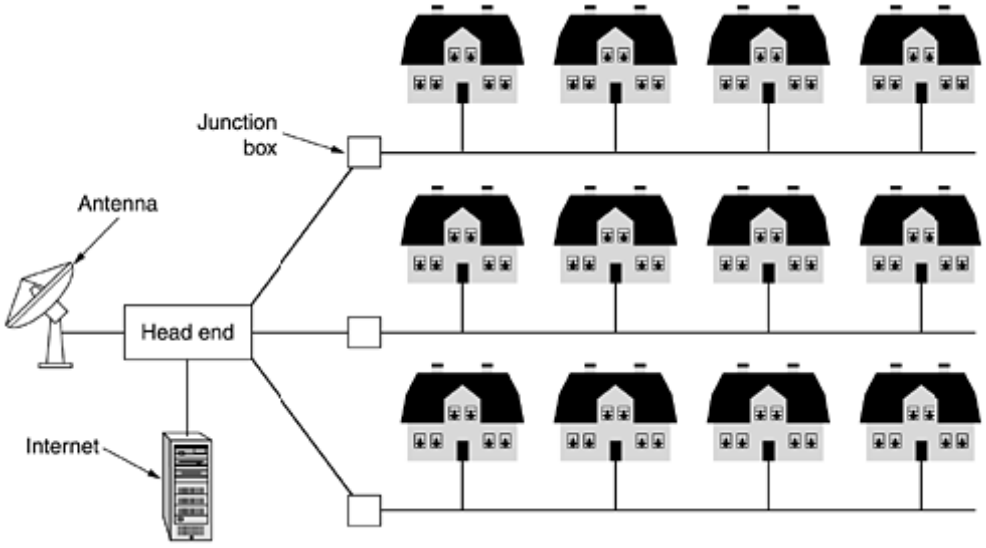

Figure 3: A Metropolitan Are Network (MAN) Based System.

CONCLUSION

Registration is the process of adding a vehicle to the Motor Vehicle Register and issuing it with registration plates. Vehicles must be registered in order to be used on the public roads and only require re-registration if the registration has been cancelled. Vehicle licensing is the payment of a fee for the use of a motor vehicle on public roads. When the fee is paid, an owner receives a label indicating the expiration date of the license. This license label must be displayed on the vehicle.

In this research, an attempt is made to design a distributed database system for automobile licensing and registration in Nigeria. The system consists of a relational database of automobile decision variables which could be shared by the three level organizations, viz, VIO, Federal Road Safety Commission and the Board of Internal Revenue. Each of these organizations is considered as a site.

The client-server architecture is used as a platform for the distributed database application development and the metropolitan area network. The essential decision variables were identified. The Database is expected to be hosted by the VIO while the concept of data replication and fragmentation is adopted by other sites to have access to data/records in the database. The system is intelligent and capable of checking to detect multiple registrations, registration of stolen automobile, malicious registration, and registration of damaged or reformed automobile and fictitious registration. The system is capable of generating reports for decision makers to enable monitoring.

REFERENCES

1. Dates, C.J. 1986. *An Introduction to Database System, Fourth Edition*. Addison-Wesley Publishing Company: Los Angeles, CA.

2. Kay-Romer, P., Pilhofer, F. and Arno, P. 2006. *Distributed System Architecture*. Morgan Kaufmann Publishers: San Francisco, CA.

3. Navathe, E. 2000. *Fundamental of Database Systems, Third Edition.* Teturo Sawada, Exclusive Publisher and Distributor.

4. Ramakrishnan, R. and Gehrke, J. 2004. *Database Managenment Systems, Second Edition*. McGraw-Hill: New York, NY. http://www.cs.wisc.edu/~dbbook

5. Tanenbaum, A.S. 2008. *Computer Networks, Foruth Edition*. Prentice Hall: Princeton, NJ. http://www.cs.vu.nl/~ast/

ABOUT THE AUTHORS

Olabode Olatubosun, Ph.D., is professional computer scientist. He obtained his B.Tech. degree in Ind. Maths degree in 1991, M.Tech. degree in computer science in 1999, and Ph.D. Degree in computer science in 2005 from the Federal University of Technology, Akure, Nigeria. Presently, he is a lecture in computer science department, Federal University of Technology, Akure, Nigeria. He has since been teaching courses and researching in the areas of information system, databases, expert systems, image processing, software engineering, data mining, advance micro-computing, probability theory, and e-commerce. Dr. Olabode has great aptitude for programming and solving numerical problems, particularly those that are related to industry. He has numerous journal publications and conference proceedings to his credit. Presently, he is a consultant to Ondo State Government and Rufus Giwa Polytechnic, Owo. He has served as a technical resource expert in many workshops organized for Local Governments in Ondo State. He is also a consultant to JT Canada. He is a member of research groups in the University where he is currently working.

Solanke Olakunle, obtained a BSc and M.Tech. degree in Computer Science and is currently pursuing a doctoral program in computer science. Presently, he works with Olabisi Onabanjo University, Ago Iwoye, Nigeria. His area of specialization includes, networking, operating systems, automata theory, and artificial intelligence.

SUGGESTED CITATION

Olatubosun, O. and O. Solanke. 2008. "Framework for Client-Server Distributed Database System for Licensing and registration of Automobiles in Nigeria". *Pacific Journal of Science and Technology.* 9(2):386-394.


Pacific Journal of Science and Technology