Designing a Hybrid Relational Database Management System (HRDBMS)

Mohammad Zaki Al_layla
College of Computer Sciences and Mathematics, University of Mosul, Mosul, Iraq

Abstract:

In this paper a hybrid relational database management system has been built, according to the client/server model of the relational SQL and by using Jet .mdb database files with Visual Basic. This system can be done by creating a Component Object Model (COM) front-ends approximates the capabilities with the performance of the client/server relational database management system at a substantially lower cost for both hardware and software.

Introduction: -

In the recent years, data is becoming more and more important for a lot of people. It is true to say that is the decade of the information overflow. The advent of wire and wireless communications and the increase developments in computer hardware posed a new challenge on the computing field. People want to use and access the information systems from their office, home, or even while they are roaming around. Portable computers, carried by different users, have changed the information access pattern. Nowadays, users don't have to stick at their offices or homes to gain access to the information systems.

Client/server database engines provide support at the server to execute SQL requests issued from the client workstation. Relational database management systems are the current technology for data management. In this paper Active X Data Objects has been used, which defined as an extensive new solution technology for the Internet. ActiveX is a broad and powerful abstraction for Microsoft Internet Solutions, and including the Access Data Object model, to verifying the connection of the sources besides relational database [1].

Component Object Model & Component Object Model ++ Programming:-

- **What is Component Object Model: -**

which means a specification for writing reusable software that allows clients and objects to communicate across process and host computer boundaries [2]. Component software development is currently one of the most prominent trends in the software industry; most of the software development tools support component software development, including Microsoft Visual Basic, Microsoft Visual C++, Delphi, and Microsoft Visual J++ [1].

Data Access Interfaces: -

A data access interface is an object model that represents various facets of accessing data. In Visual Basic, three data access interfaces are available: -

1. ActiveX Data Objects (ADO).
2. Remote Data Objects (RDO).
3. Data Access Objects (DAO).

Using Visual Basic can programmatically control connecting to the database, retrieving records, and changing the value of records. Any of the three data access technologies can be used to interact with a database; ActiveX Data Object is the newest and most powerful. Because ActiveX Data Objects is an interface to Object Link Embedded Database, Microsoft's newest and most powerful data access technology, ActiveX Data Objects provides high-performance access to a variety of information sources (including relational data and nonrelational data). This includes mainframe indexed sequential access method/virtual storage (ISAM)/(VSAM), hierarchical databases, desktop database such as Microsoft Access, and remote database such as Oracle and Microsoft Structured Query Language Server. In addition, ActiveX Data Objects can access other data such as e-mail servers, file system stores, text files, graphical and geographical data, custom business objects, and more [3].

In ASP.NET and Visual Basic.net the data source controls corresponding ActiveX Data Object in the Visual Basic manage the tasks of connecting to a data source and reading and writing data. Data source controls do not render any user interface, but instead act as an intermediary between a particular data store (such as a database, business object, or XML file) and other controls on the ASP.NET Web page. Data source controls enable rich capabilities for retrieving and modifying data, including querying, sorting, paging, filtering, updating, deleting, and inserting [7].

Data Access Objects (DAO): -

The first data access technology introduced in Visual Basic was Data Access Object (DAO). Data Access Object lets access and manipulate data in local stone alone computer or remote databases and manage the structure of certain types of databases (client/server computer). Data Access Object provides a hierarchical object model, which makes using Data Access Object in easy. Data Access Object supports two basic ways to access data. Choose any of these ways based on the type of database application is relating to.

a. Microsoft Joint Engine Technology (Jet) allows to access data in desktop data sources, such as Microsoft Access, FoxPro, Paradox, or Lotus 1-2-3, this consider as the first way to access the data.

b. Open Database Connectivity (ODBC) Direct allows accessing remote database servers without using the Microsoft Jet database engine. This provides better performance and also requires less memory, and considers the second way to access the data.

- **Remote Data Objects (RDO): -**

Provides an object model for accessing remote data. The RDO programming model is similar to the DAO model, except that it is designed to work with client/server databases rather than desktop databases. RDO takes advantage of intelligent database servers that use sophisticated query engines, such as SQL Server and Oracle.

- **ActiveX Data Objects (ADO): -**

Microsoft's newest data access technology and is an interface to Object Link Embedded Database (OLE DB) (OLE DB: A specification for a set of data access interfaces

designed to enable a multitude of data stores, of all types and sizes, to work seamlessly together. These interfaces comprise an industry standard for data access and manipulation that can ensure consistency and interoperability in a heterogeneous world of data and data types [5].), where Object Link Embedded Database and Active X Data Objects provide developers the same interface not only access data from relational and nonrelational databases, but also other data sources, such as e-mail, file systems, project management tools, spreadsheets, and custom business objects. OLE DB has been designed to build on the success of ODBC by providing an open standard for accessing all kinds of data. Because of OLE DB, so can build solutions that span desktop, midrange, mainframe, and Internet technologies using a variety of data stores. The reasons behind using Visual Basic with Component Object Model is that Visual Basic offers the highest levels of productivity. Visual Basic 3.0 and Visual Basic 4.0 both offered modest advancements in the product's Component Object Model awareness, but Visual Basic 5.0 really opened the door to make this development tool a viable option for building components for use in Component Object Model based systems. Visual Basic 6.0 enterprise continues to add to the Component Object Model capabilities of this development tool (1), so Microsoft Visual Basic 6.0 enterprise allows utilizing Component Object Model through interface-based programming, and makes it relates to the Component Object Model, which that used in this paper. Because Visual Basic developer permits to create Component Object Model components with ActiveX controls such as Active X Data Objects, ActiveX Documents, ActiveX EXE, and ActiveX DLL projects, and because Visual Basic was designed to provide developers with a high level of productivity and rapid application development, Visual Basic hides most of the complexities of Component Object Model [2].

Interface-Based Programming: -

It is a concept created by the computer science academia and companies that needed to develop enterprise software and information systems. An interface is a distinct data type. It defines a set of public methods without including any implementation. In another sense, an interface defines a very specific protocol for the communications that occur between a client and an object [(1)]. Conceptually, interface-based programming defines two separate but related elements: an interface and a coclass. An interface is a data type that contains definitions of public methods without containing any code. The coclass contains the code for the methods defined by the interface, as illustrated in Figure1 [2].

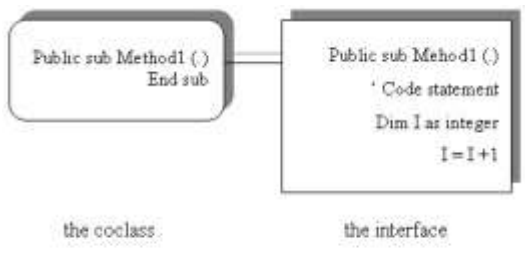

Figure 1: the relational ship between the interface and the coclass

The interface contains the declaration of method1 but does not contain any code. The coclass contains code defined by the interface. Component Object Model provides the ability for objects to communicate with each other both in-process and out-of-process. Out-of-process objects can be located either on the same computer or different computers on a network. Interface pointers achieve this communication. For example, when a client application calls a method of a Component Object Model object, the client points to a location in memory where the COM object's interface resides, as illustrated in Figure 2 [2].

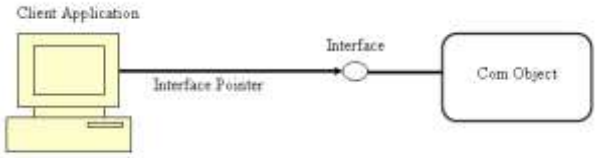

Figure 2: the ability of COM to communication with different computers on the network

Building the database System: -

After designing the creation of the schema, the database can be created by using the Microsoft Access, which is easier-to-use interface for creating database objects. Because Visual Basic 6.0 enterprise and Microsoft Access 97 share the same database engine, so it can use either Visual Basic or Microsoft Access to create the database; the resulting database file created with two systems are identical. So the Microsoft Access has been used to build the database and its tables, entering the information at each table. And then the interfacing used programming by Visual Basic, to view the data of the database, making the right operations such as update, delete, create new, etc.

- **Configuring & Open Database Connectivity (ODBC): -**

Open Database Connectivity is windows technology that lets a database client application connect to a sever database.

The definition of the Open Database Connectivity is client-side technology that seeks every, the server (back-end) a data source generic. Open Database Connectivity is composed of three parts[4]

a. A drive manager.
b. One or more drivers.
c. One or more data source.

The architecture of Open Database Connectivity is shown in the following figure 3.

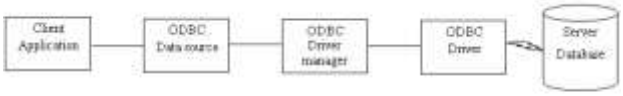

Figure 3: Open Database Connectivity architecture showing the connection Between the client & the server database

- **Creating an Open Database Connectivity Data Source: -**

Before creating a client application to connect to a client/server database and using Open Database Connectivity, there must be first supply information about the Open Database Connectivity data source on the client, such as the user name, the password, and others, which give client application the capability to refer to a

combination of the Open Database Connectivity driver and the database.
To create an Open Database Connectivity data source name on the client, there are several things which must be taken into consideration [4].

- **Creating Active X Data Objects with Visual Basic 6.0 Enterprise: -**

Universal Data Access (UDA) is the Microsoft strategy for providing access to all types of information, from a variety of sources besides relational databases. These data sources include mainframe, hierarchical databases, email, file systems, text, and graphical data [2].
The Microsoft Universal Data Access strategy is that delivered through a common set of object-oriented interfaces. These interfaces are based on the Microsoft Component Object Model. Object Link Embedded Database (OLE DB), which it is technology and interface for object interaction, a way to transfer and share information between programs, and it is based on Component Object Model, used to provide access to data across the organization, as illustrated in Figure 4.

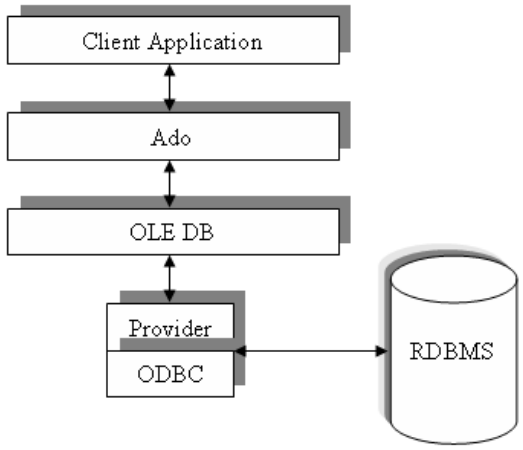

Figure 4: the universal data access architecture

The Active X Data Objects object model provides an easy-to-use set of objects, properties, and methods for creating applications that access and manipulate data. The architecture of the Active X Data Objects is shown

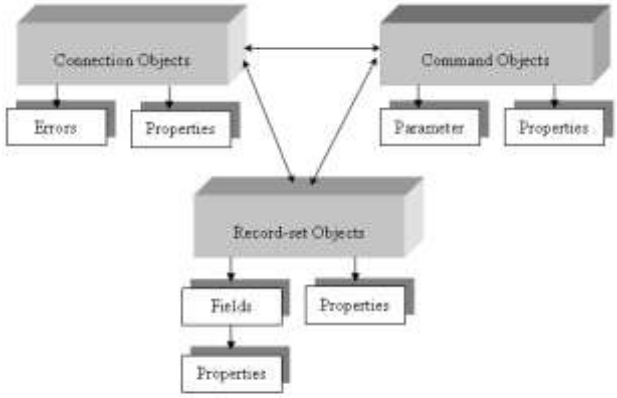

Figure 5: The Ado Object Model

in the following figure 5 [2].

1. Connection object, which maintains connection information with the data provider. Error object, which contains extended error information about a condition raised by the provider.
2. Command object, which maintains information about a command. Parameter object, which it is single parameter for a parameterized command.
3. Recordset object, which contains a set of records, returned from a query.Field object, which contains information about a single column of data within a recordset. And finally Property object, which contains a provider-defined characteristic of an Active X Data Objects.

The Connection object establishes a connection to a data source (server). It allows application to pass client information to create a connection, i.e. before establishing a connection; an application can create a connection string that includes the user's logon name and password. In the interface programming it is used to establish the connection through the client computer, and must type the name and the password to enter to the database files.
Command objects define specific detailed information about what data is retrieved from a database connection. Command objects can be based on either a database object (such as a table, view, or stored procedure) or a Structured Query Language (SQL) query. A Recordset object represents the entire set of records from a database table or the results of an executed command such as adding a new customer, updating its information deletion and etc. All Recordset objects are constructed using records (rows) and fields (columns).

Conclusions: -

1. Justify the building of a hybrid relational database management system, according to the client/server model of the relational SQL and using
Jet .mdb database files with Visual Basic developer.
2. Learning about how to write reusable software that allows clients and objects to communicate across process and host computer boundaries.
3. Develop enterprise software and information systems.
4. Programming the coclass to define very specific protocol for the communications that occur between a client and an object.

References: -

1. Library of Congress Cataloging-in-Publication Data (1999) “Distributed Applications with Microsoft Visual Basic 6.0 MCSD Training Kit“, Microsoft Corporation, United States of America.
2. Library of Congress Cataloging-in-Publication Data (1998) “Programming Distributed Applications with COM and Microsoft Visual Basic 6.0/Ted Pattison”, Microsoft Corporation, United States of America
3. Library of Congress Cataloging-in-Publication Data (1999): “Desktop Applications with Microsoft Visual Basic 6.0 MCSD Training Kit”, Microsoft Corporation, United States of America.
4. Jeffrey P. McManus (1998): “Database Access with Visual Basic 5.0”, Sams, CA.

5. David Garrett, et. al. (1996): "Intranets Unleashed, First Edition", IntraACTIVE, Inc., United States of America.
6. Roger Sessions, (2001): "Debug Visual Basic.Net", Microsoft's Vision for Distributed Objects, published by john & sons, Inc, Canada.
7. David G. Jung Jeff Kent, (2000): "COM and DCOM", The McGraw-Hill companies, United States of America. .

تصميم نظام قاعدة البيانات للعلاقائية الادارية المولدة (المهجنة)

محمد زكي الليلة

كلية علوم الحاسبات والرياضيات، جامعة الموصل، الموصل، جمهورية العراق

المخلص:

في هذا البحث تم بناء نظام قاعدة البيانات للعلاقائية الادارية المهجنة وحسب منهج الزبون/خادم ذو العلاقائية الاداري (SQL) باستخدام ملفات قاعدة البيانات (MBD) مع مطور الرؤية الاساسي. ويمكن تنفيذ النظام بتكوين منهج نظام قاعدة البيانات تقرب الامكانيات باداء منهج نظام قاعدة البيانات للعلاقائية الادارية لـ الزبون/خادم بكلفة متدنية بشكل كبير للبرامجيات والاجزاء الملموسة.