

THE ENCRYPTION – DECRYPTION OF THE COLUMN LEVEL FOR A  
COMMERCIAL DBMS WITH SUPPORTING USER INTERFACE

ELFADIL SABEL MOHAMED ALI

UNIVERSITI TEKNOLOGI MALAYSIA

THE ENCRYPTION – DECRYPTION OF THE COLUMN LEVEL FOR A  
COMMERCIAL DBMS WITH SUPPORTING USER INTERFACE

ELFADIL SABEIL MOHAMED ALI

A project submitted in fulfilment of the  
requirements for the award of the degree of  
Master of Computer Science (Information Security)

Centre for Advanced Software Engineering (CASE)  
Faculty of Computer Science and Information System  
Universiti Teknologi Malaysia

APRIL 2009

To my beloved parents, wife, daughters, brothers and sisters

## ACKNOWLEDGEMENT

All praise be to Allah, the Most Merciful, for His Love and Guidance. Salutations on the Prophet Muhammad (PBUH), his family, and fellow companions.

May I express my appreciation to ALLAH, the beneficent, the merciful, for making me a Muslim and blessing me with the privilege of acquiring a higher degree. My heartfelt gratitude goes to my parents for bearing with me weakness upon weakness from cradle to date.

I would like to thank my supervisor, Assoc. Prof. Dr. Zailani Mohamed Sidek, for his direction, time, and motivation, throughout the project. I wish to thank my colleagues Ahmed, Saeed, Abdalaleem, Haniza, Neema, Mesam, Hamed and Ala Aldeen and others for their support and encouragement. May ALLAH reward you all the relentless efforts to see through this academic pursuit.

## **ABSTRACT**

The main purpose of the project was to develop an improvement of the database column level encryption implementation. The Small and medium size enterprises (SMEs) need to secure their valuable database from intrusion by unauthorized personnel economically. Nevertheless, they have identified a need for improvement to their database column level encryption, in order to enable a more efficient and simple approach to the maintenance and securing of the data. This project attempts to build an appropriate solution that would enable the System manager to create system users and give them roles. The security manger makes encryption to the column database only, while only the data owner can decrypt the data column. The project also gives a background and a detailed explanation of the Database Access Technologies (DATs) such as ODBC, JDBC, XML and .NET etc.; and how they were used in relation to this project. Emphasis was placed on content management to develop a user friendly Graphical User Interface. Some different software used during this project: Microsoft Windows Vista, SQL Server to host the database and Microsoft.NET (VB.NET) as the front-end. Microsoft.NET (VB.NET) is the middleware used to connect Operating System, Graphical User Interface and SQL Server database. The Secure Column Application is the product of the above mentioned project and it provides high security for MS SQL Server 2005 and presents transparent encryption and decryption with any type of the database table column.

## ABSTRAK

Tujuan utama projek ini adalah untuk membina peningkatan kepada pelaksanaan tahap enkripsi kolum pangkalan data. Perusahaan kecil dan sederhana (SMEs) perlu mempertahankan pangkalan data yang bernilai daripada pencerobohan personal yang tidak dibenarkan secara ekonomik. Malahan, mereka telah mengenalpasti keperluan bagi meningkatkan tahap enkripsi kolum pangkalan data, dalam kearah pendekatan yang lebih efisien dan mudah bagi tujuan penyelenggaraan serta mempertahankan data. Projek ini berusaha untuk membangunkan penyelesaian yang bersesuaian untuk membenarkan Pengurus Sistem mencipta dan memberi peranan kepada pengguna-pengguna sistem. Pengurus Keselamatan melaksanakan enkripsi kepada kolum pangkalan data sahaja, sementara itu hanya pemilik data dibolehkan melaksanakan deskripsi kolum data tersebut. Projek ini juga memberi gambaran dan penjelasan terperinci mengenai Database Access Technologies (DATs) seperti ODBC, JDBC, XML and .NET dan sebagainya; dan bagaimana ia digunakan secara lansung dalam projek ini. Penekanan diletakkan ke atas pengurusan maklumat dalam membangunkan antaramuka mesra pengguna Graphical User Interface. Antara perisian berlainan yang digunakan sepanjang projek ini: Microsoft Windows Vista, SQL Server sebagai hos pangkalan data dan Microsoft.NET(VB.NET) sebagai bahagian depan. Microsoft.NET (VB.NET) sebagai 'middleware' digunakan untuk menghubungkan Sistem Pengoperasian, Graphical User Interface dan Pangkalan Data SQL Server. Secure Column Application adalah produk yang dimaksudkan di dalam projek ini dan ia membekalkan keselamatan tinggi kepada MS SQL Server 2005 serta mempersembahkan ketelusan enkripsi dan dekripsi dengan apa jua jenis kolum jadual pangkalan data.

## TABLES OF CONTENTS

CHAPTER	TITLE	PAGE
	<b>DECLARATION</b>	ii
	<b>DEDICATION</b>	iii
	<b>ACKNOWLEDGEMENT</b>	iv
	<b>ABSTRACT</b>	v
	<b>ABSTRAK</b>	vi
	<b>TABLE OF CONTENTS</b>	vii
	<b>LIST OF TABLES</b>	xiv
	<b>LIST OF FIGURES</b>	xv
	<b>LIST OF ABBREVIATIONS</b>	xvii
	<b>LIST OF APPENDICES</b>	xix
<b>1</b>	<b>PROJECT OVERVIEW</b>	<b>1</b>
	1.1 Introduction	1
	1.2 Background	2
	1.3 Statement of the Problem	3
	1.4 Objectives	3
	1.5 Scope Of Study	4
	1.6 Importance of The Study	4
	1.7 Summary	4
<b>2</b>	<b>LITERATURE REVIEW</b>	<b>6</b>
	2.1 Introduction	6
	2.2 Database	7
	2.2.1 What Is Database?	7
	2.2.2 What Is Database Management System	8

	(DBMs)?	
	2.2.3 Database Security	8
	2.2.4 Database Cryptography	9
	2.2.5 Database – Level Encryption	10
	2.2.6 SQL Server And Database Encryption Keys (Database Engine)	11
	2.3 Database Access Technologies (DATs)	11
	2.3.1 Open Data Base Connectivity (ODBC)	12
	2.3.2 Java Database Connectivity (JDBC)	19
	2.3.3 ActiveX Data Objects (ADO)	23
	2.3.4 Online Analytical Processing (OLAP)	25
	2.3.5 Object Linking And Embedding (OLE)	26
	2.3.6 CORBA	28
	2.3.7 ADO.NET	31
	2.3.8 Extensible Markup Language (XML)	33
	2.3.9 Microsoft.NET	37
	2.4 Summary	38
<b>3</b>	<b>RESEARCH METHODOLOGY</b>	<b>39</b>
	3.1 Introduction	39
	3.2 The Research Development Phases	39
	3.2.1 Project Preview	40
	3.2.2 Literature Review	41
	3.2.3 Studies and Analysis	41
	3.2.4 Determine the research requirements	41
	3.2.4.1 Software Development Tools	42
	3.2.5 System Design	44
	3.2.5.1 Database design	44
	3.2.6 System Implementation and Testing	45
	3.2.7 Benefits, Discussion and Future Works	46
	3.3 Summary	46
<b>4</b>	<b>STUDIES AND ANALYSIS'S</b>	<b>47</b>



4.1	Introduction	47
4.2	The Java Database Connectivity (JDBC)	47
4.2.1	JDBC API Overview	47
4.2.2	JDBC Architecture	48
4.2.2.1	Type 1 Driver - JDBC-ODBC bridge	48
4.2.2.2	Type 2 Driver - Native-API Driver specification	50
4.2.2.3	Type 3 Driver - Network-Protocol Driver	51
4.2.2.4	Type 4 Driver - Native-Protocol Driver	54
4.2.3	Comparison of the 4 Types Of JDBC Drivers	57
4.2.4	General Advantages of JDBC Technology	58
4.3	Microsoft Open Database Connectivity (ODBC)	59
4.3.1	How ODBC Works?	61
4.3.2	ODBC Drivers	62
4.3.2.1	Driver Tasks	63
4.3.2.2	Driver Architecture	63
4.3.2.3	Other Driver Architectures	67
4.3.2.4	Microsoft ODBC Desktop Database Drivers	69
4.3.2.5	ODBC Driver For Oracle	70
4.3.2.6	Oracle Form Developer And ODBC	70
4.3.2.7	Visual FoxPro ODBC Driver	71
4.3.2.8	ODBC Architecture	71
4.3.3	Encryption	72
4.3.4	ODBC Advantages	72
4.3.5	ODBC Disadvantages	73
4.4	ActiveX Data Objects (ADO)	74
4.4.1	ADO Fundamentals	75
4.4.2	ADO Advantages	77

4.4.3	ADO Disadvantage	77
4.5	ADO.NET	77
4.5.1	ADO.NET Architecture	78
4.5.2	ADO.NET Encryption	80
4.5.3	ADO.NET Advantages	80
4.5.4	ADO.NET Disadvantages	81
4.6	The Common Object Request Broker Architecture (CORBA)	81
4.6.1	Interface Definition Language (IDL)	83
4.6.2	How is CORBA works	83
4.6.3	CORBA Features	85
4.6.4	Encryption	85
4.6.5	CORBA advantages	85
4.6.6	CORBA Disadvantages	87
4.7	Online Analytical Processing (OLAP)	88
4.7.1	OLAP Design	90
4.7.2	OLAP Security	90
4.7.3	OLAP Advantages	91
4.7.4	OLAP Disadvantages	91
4.8	OLE	92
4.8.1	OLE DB Programming Overview	93
4.8.2	Universal Data Access	93
4.8.3	Benefits of COM	94
3.8.4	Drawbacks of COM	95
3.8.5	OLE DB Architecture	95
3.8.6	How OLE Works	95

4.8.7	Oracle Objects for OLE (OO4O)	96
4.8.8	OLE Advantages	96
4.8.9	OLE disadvantages	97
4.9	XML	98
4.9.1	Features / capabilities	99
4.9.1.1	Oracle	99
4.9.1.2	Microsoft SQL Server 2005	100
4.9.2	How XML Schema Processing with SQL server 2005?	100
4.9.3	Encryption	101
4.9.4	XML Advantages	102
4.9.5	XML Disadvantages	104
4.10	Microsoft.Net	105
4.10.1	What is the .NET Framework and Visual Studio?	105
4.10.2	General Goals of .NET	106
4.10.3	.NET Architecture	107
4.10.4	Principal Design Features	110
4.10.4.1	General features	110
4.10.4.2	ADO.NET Entity Framework	111
4.10.4.3	What's New in the .NET Framework Version 3.5 SP1	112
4.10.5	Data Encryption In SQL Server (ADO.NET)	113
4.10.6	Visual Basic.NET	114
4.10.6.1	Simple	115

4.10.6.2	Funny	115
4.10.6.3	Easy to Learn	116
4.10.7	Advantages of Using Managed Code to Create Database Objects	116
4.10.8	Disadvantages of .NET	119
4.11	Comparative Studies	120
4.12	Recommendations	127
4.13	Summary	129
<b>5</b>	<b>SYSTEM DESIGN</b>	<b>130</b>
5.1	Introduction	130
5.2	User Requirements	130
5.2.1	Initialization	131
5.2.2	Login	131
5.2.3	Main Form	132
5.2.4	Manage User Information	133
5.2.5	Encryption	133
5.2.6	Decryption	133
5.3	Database Design	134
5.3.1	Dbo.UserInfo	134
5.3.2	Scolumn.Sec_Useraccessinfo	135
5.3.3	Scolumn.Sec_Securityinfo	135
5.3.4	Scolumn.Sec_Cryptoinfo	136
5.4	System Requirements	136
5.4.1	Hardware Requirement Specification	136
5.4.2	Software Requirement Specification	137
5.5	Summary	137
<b>6</b>	<b>SYSTEM IMPLEMENTATION AND TESTING</b>	<b>138</b>
6.1	Introduction	138
6.2	Coding Steps	138
6.2.1	Initialization	138

6.2.2	User and key management unit:	139
6.2.3	Test and Evaluation the result	140
6.3	Administrator and User Manual	141
6.3.1	Create users by System Administrator	141
6.3.1.1	Run the SQL Server 2005 Express Edition	141
6.3.1.2	How to make Initialization	145
6.3.1.3	How to make Encryption	148
6.3.1.4	How to make Decryption	151
6.4	Summary	152
<b>7</b>	<b>BENEFITS, DISCUSSION AND FUTURE WORKS</b>	<b>153</b>
7.1	Introduction	153
7.2	Implemented Modules	153
7.3	Lessons Learnt	155
7.4	Expected Organizational Benefits	155
7.5	Future Works	157
7.6	Summary	157
	<b>LIST OF REFERENCES</b>	<b>159</b>
	<b>APPENDICES A-E</b>	<b>167-169</b>

**LIST OF TABLES**

<b>TABLE NO.</b>	<b>TITLE</b>	<b>PAGE</b>
4.1	JDBC Drivers types contrast	57
4.2	Comparison between DATs Capabilities	120
4.3	Pros and Cons of Data Access Technologies	121
4.4	J2EE/JDBC vs .NET/VB.NET	127
5.1	DBO.USERINFO table design	134
5.2	SCOLUMN.SEC_USERACCESSINFOTABLE DESIGN	135
5.3	SCOLUMN.SEC_SECURITYINFO TABLE DESIGN	135
5.4	SCOLUMN.SEC_CRYPTOTOINFO TABLE DESIGN	136
5.5	The Hardware Specifications	137
6.1	Test Approaches of Secure Column Application	140
6.2	Test Result of Secure Column Application Test	140

## LIST OF FIGURES

FIGURE NO	TITLE	PAGE
2.1	Flow of project data	6
2.2	Web Services-based access to relational databases	22
3.1	Research Flow Chart	40
3.2	The overall Project Layout	44
4.1	Schematic of the JDBC-ODBC Bridge	49
4.2	Schematic of the Native API driver	51
4.3	Schematic of the Network Protocol driver	52
4.4	Schematic of the Native-Protocol driver	54
4.5	Data flows over the network	56
4.6	Different configurations of file-based drivers	64
4.7	Different configurations of DBMS-based drivers	66
4.8	Single network configuration	67
4.9	Installation of ODBC on the Server	68
4.10	ADO.NET architecture	80
4.11	The structure of ORB interface	83
4.12	A component DBMS architecture using OLE DB interfaces	96
4.13	Visual Studio overview of the Common Language Infrastructure	109
4.14	The Structure of Microsoft.Net	109
4.15	Microsoft Visual Studio (VB.NET)	114
5.1	Encryption Main Form	132
5.2	Decryption Main Form	132
6.1	Run SQL Server 2005 Express Edition	141
6.2	Server connection by System Administrator	142

6.3	Management Options	142
6.4	Create New Login	143
6.5	Login-New	144
6.6	Create New user on Database.	144
6.7	Add user name and Login name	145
6.8	Log Status File	145
6.9	System Administrator Authentication	146
6.10	Automatic database detection	146
6.11	Admin's Login Account	147
6.12	Initialization successful	147
6.13	Dbp.userinfo table before encryption	148
6.14	Database Login authentication	148
6.15	Encryption System Login authentication	149
6.16	Display of all databases automatically	149
6.17	The data before encryption.	150
6.18	Columns Encryption.	150
6.19	Data user Login to the Database	151
6.20	Data user Login to the Encryption System.	151
6.21	Columns decryption Result.	152



## LIST OF ABBREVIATIONS

ADO	-	Activex Data Objects
ADO.NET	-	Activex Data Objects .NET
AMI	-	Asynchronous Method Invocation
API	-	Application Programming Interface
ASP.NET	-	Active Server Page. Net
ATM	-	Asynchronous Transfer Mode
BCL	-	Base Class Library
BPM	-	Business Process Management
CAS	-	Code Access Security
CCA	-	Common Component Architecture
CIA	-	Confidentiality, Integrity And Availability
CLI	-	Call Level Interface
CLR	-	Common Language Runtime
COM	-	Component Object Model
CORBA	-	Common Object Request Broker Architecture
CPU	-	Central Processing Unit
DAO	-	Data Access Objects
DATs	-	Database Access Technology
<i>DB</i>	-	Database
DBA	-	Database Administrator
DLL	-	Dynamic Link Library
DM	-	Data Mining
DMF	-	Decryption Main Form
DTDs	-	Document Type Definitions
ECMA-335	-	European Computer Manufacturers Association
EMF	-	Encryption Main Form
FCL	-	Framework Class Library

GUIs	-	Graphical User Interfaces
ICTs	-	Information And Communication Technologies
IDE	-	Integrated Development Environment
IDL	-	Interface Definition Language
IETF	-	Internet Engineering Task Force
IIOp	-	Internet Inter-Orb Protocol
IIS	-	Internet Information Services
IMS	-	Instrumentation Maintenance System
ISVs	-	Independent Software Vendors
IVs	-	Initialization Vectors
JDBC	-	Java Database Connectivity
JIT	-	Just-In-Time
JNDI	-	Java Naming And Directory Interface
JVM	-	Java Virtual Machine
LINQ	-	Language-Integrated Query
MDAC	-	Microsoft Data Access Components
MDX	-	Multidimensional Expressions Or
MFC	-	Microsoft Foundation Classes
MOLAP	-	Multidimensional Online Analytical Processing
.NET	-	Microsoft.NET
OASIS	-	Organization For The Advancement Of Structured Information Standards
OCA	-	Oracle Open Client
ODBC	-	Open Database Connectivity
OLAP	-	Online Analysis Process
OLE	-	Object Linking And Embedding
OLTP	-	Online Transaction Processing
OMG	-	Object Management Group
OO	-	Object Oriented
OO4O	-	Oracle Objects For OLE
OPC	-	OLE For Process Control
ORB	-	Object Request Broker
PE	-	Portable Executable

QA	-	Quality Assurance
QoS	-	Quality Of Service
RAD	-	Rapid Application Development
RDO	-	Remote Data Objects
RMI	-	Remote Method Invocation
ROLAP	-	Relational Online Analytical Processing
RPC	-	Remote Procedure Calling
SCL	-	Service Control Language
SDK	-	Software Development Kit
SGML	-	Standard Generalized Markup Language
SMEs	-	Small And Medium Size Enterprises
SProc	-	Stored Procedure
SQL	-	Structure Query Language
SSL	-	Secure Sockets Layer
TCP/IP	-	Transmission Communication Protocol /Internet Protocol
T-SQL	-	Transact-SQL
UDA	-	Universal Data Access
UDF	-	User Defined Functions
UI	-	User Interface
UPC	-	Universal Personal Computing
VB	-	Visual Basic )
VB.NET	-	Visual Basic .Net
VBA	-	Visual Basic For Application
VPN	-	Virtual Private Network
W3C	-	World Wide Web Consortium
WCF	-	Windows Communication Foundation
WPF	-	Windows Presentation Foundation
XKMS	-	Xml Key Management Specification
XML	-	Extensible Markup Language

**LIST OF APPENDICES**

<b>APPENDIX</b>	<b>TITLE</b>	<b>PAGE</b>
A	Figure 1 Partial Column Encryption Process	167
B	Figure 2: Partial Column Decryption Process	167
C	Figure 3: Partial Column Encryption Result	168
D	Figure 4: Partial Column Decryption Result.	168
E	Figure 5: Project Ghant Chart	169

## CHAPTER 1

### PROJECT OVERVIEW

#### 1.1 Introduction

Nowadays, the developments in information and communication technologies (ICTs) have a great impact to all sectors of private and public life. These manifest in database and information processing by transacting with the computer systems. For instance, data is manipulating to generate or produce useful products or services to serve banking the sector, transport industry, communication system, industrial development and the organization human resources. All these are very dependent on the scheme. Furthermore, these organizations rely on the system to store and back up a huge amount of data. Consequently, they offer appropriate services of accelerated transactions as and when required. Generally, the database and database management (DBAs) consider as the heart of the information system applications.

Although the using of the database and computer technology system contributes in the development of organizational service delivery, information security threats arise. Such threats and breaches that internally or externally generated or prompted by malicious. Accordingly, insurance of effective working of database and information security requires various controls and measures handle in information security management systems. The aim is to maintain data confidentiality, integrity and availability (CIA), such as polices, antivirus software, access control, and cryptography. However, lack of proper security control implementation and management could lead to a disaster within an organization. Undoubtedly, the most important of these controls is the ‘last controls defense’

(cryptography). Hence, this project examines the database security level, in part of confidentiality of the column contents, in addition to, the usability of the user interface.

## 1.2 Background

The most challenging of the database information systems are threats. Threats define as a source of potential harm that may cause damage to the system, through exploit vulnerabilities. They classified into two, physical and logical threats (Zailani, 2004). Logical threats or malicious attacks know as unauthorized access to information in database via software. They may cause information disclosure, loss intact or integrity and inaccessibility to authorized users. However, these kinds of attacks may cause alteration and disclosure to information stored in database.

Usually, the database system provides some security controls to meet most requirements, such as constraints, access control, and authenticity and so on. Nevertheless, data secrecy is still exposing to internal hacking by unauthorized and authorized users like database administrator (DBA).

Writing codes in structure query language (SQL) prompt does all transactions subject to the following competencies: -

1. SQL design statement
2. Using and writing INSERT statements
3. Using and writing SELECT statement,
4. Query only data in his authorization

In addition, for confidentiality, integrity and availability, the DBA has to write some codes when additional security required (see appendix A-D). Moreover, all the users must know how to deal with SQL prompt when writing, updating and querying. Finally, Database Crypto system that provides encryption and decryption

to the column level implement through securely, humane and friendly user interfaces is required.

### **1.3 Statement of the problem**

In response to security threat to businesses' computer system, cryptographic application, the last defense system is inevitable. Securing sensitive and critical database demands compassionate and effective key management of an encrypted system. Small and medium size enterprises (SMEs) need to secure their valuable database free from intrusion by unauthorized personnel economically. Hence, a comprehensive security system that promotes system users interface interaction is quite desirable. The basic issues are as follows: -

*How to recognize the database (DB) table automatically and allow users to choose encryption/decryption columns?*

### **1.4 Objective**

The objectives of this project are as follows: -

- i) To review Database Access Technologies in order to recommend suitability their application.
- ii) To use Database Access Technology for automatic recognition of db tables; and
- iii) To improve the implementation of the Encryption/Decryption capabilities on the database column level, through Database Access Technology (DATs).

## **1.5 Scope of study**

This project will focus on the development of the following: -

- i) Perform Security against internal attacks to the organization data (confidentiality only).
- ii) Focus on encryption of database column level (Oracle RDBMS or SQL Server, etc...).
- iii) User Interface for data manipulation. The User Interface should work on client/server.

## **1.6 Importance of the study**

This project aims at extending organizations' database security to meet the following challenges: -

- i) Ensure confidentiality by protecting the data from unauthorized exposure.
- ii) Support the data Integrity from updating, deletion, changing of the data type by the authorized users.
- iii) Provide security of the database against internal attacks
- iv) To sustain access control mechanism through encryption scheme Provide user-friendly environment.

## **1.7 Summary**

Information and Communication Technologies (ICTs) offers great opportunities to organizations (privates, publics) to improve their products and services, as well as maintain the quality. However, lack of proper security control would impede all efforts exerted in meeting these objectives. Failure to maintain a comprehensive security system may jeopardize a businesses' critical data via threats



by hackers. Moreover, poor cryptographic key management could lead to violation of security system. Therefore, there is a need for proper functional, friendly security management system.

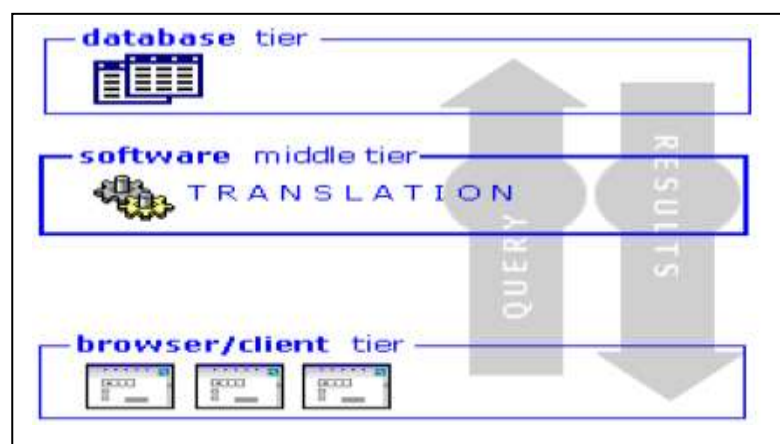
## CHAPTER 2

### LITERATURE REVIEW

#### 2.1 Introduction

In this chapter a literature review concentrates on Database (database definition, database security, database cryptography and database-level encryption). In addition to Database Access Technologies (ODBC, JDBC, ADO, OLAP, OLE, CORBA, ADO.NET, XML and Microsoft.NET) are reviewed.

Figure (2.1) below depicts the architecture required to deliver the required functionality. It shows that the browser / client front-end connect to the database through a middleware.



**Figure(2.1):** Flow of project data

## 2.2 Database

### 2.2.1 What is Database?

A database is a collection of records, data or information that can be stored, modified or retrieved at any time. The essence of a database is to have an organised method of easy access to the required information which is kept in the database. Having a database will enable businesses to manage and update data details easily, effectively and with accuracy. Databases are widely use around the world by many small, medium and large organizations. It has become an essential asset for a business to grow and have a well organised business structure.

There are many popular databases some of which as described here:-

- **Microsoft SQL Server** is a universally used database product and provides core support for Extensible Mark-up Language (XML) and Internet queries.
- **Oracle Database** is one of the leading relational database management systems; Oracle is a tool for creating Dynamic e-Business applications based around the Oracle database.
- **Microsoft Access** is an influential visual tool that can be used to design and develop Windows based database applications. Microsoft Access can also be use as a front end application base because it eases of use and flexibility (Microsoft).
- **MySQL** is a program that manages databases. It is used to achieve tasks inside a database like all the other databases and it is not the same as structured query language (SQL by IBM). It is a database that uses SQL to control, create and show data.
- **Microsoft FoxPro** is a tool for building database solutions. It is an object-oriented language.
- **IBM-DB2** database software is used around the world in large industries. It has leading capabilities in reliability, performance, and scalability.

### **2.2.2 What is database management system (DBMS)?**

Database management system is software designed to assist in maintaining and utilizing large collection of data, and the need for such systems, and their uses (Ramakrishan, 2005). Consequently, the successful operations of the computer information system demands system security and system usability (user interface - UI). Moreover, system security is concerned with the presentation of Confidentiality, availability and integrity (CIA) within the organization. The CIA refers to confidentiality of data availability only to authorized users, i.e. privacy proof and satisfaction. Integrity ensures the accuracy of data validity for as long as possible, i.e. data intact. Availability ensures data is available to authorized persons as and when required. Moreover, security experts believe that more security needs are required especially to safeguard against internal attacks, such as Identification, Authentication, Access Controls and Encryption (which is the last defense of the security). Furthermore, computer information system has adequate usability and UI, if has an environment that can assist the user to do what is needed and is easy to use, as detailed later (Lausesen, 2005).

### **2.2.3 Database Security**

A database security could be defined as is a process that issues a group of measures, rules, polices and mechanisms. In cryptographic technology, database security is the operation that comprises a security scheme for implementing a specific security policy in database to prove the CIA ( Zailani, 2004). Database security allows the organization to protect the data and combat the threats from insiders and outsiders intentionally or unintentionally. However, mismanagement of the security system could lead to serious consequences.

What's more, physical security and logical security must be prepared, implemented and monitored properly. Security requirements for database systems are as follows ( Zailani, 2004): -

- Physical database security – data can be recovered when physical problems occurred like power failure.
- Logical database security – the structure of the database is maintained.
- Integrity – all the data stored for each element is correct or accurate.
- Auditing – a process of tracking any access can be done.
- Access control – only authorized users can access the database system with identified privilege.
- Authentication - in order to ensure the right user is identified by auditing his trailer for permissions to access the system.
- Availability – user can access the database and data normally within his authority.
- Encryption – to ensure that only authorized users can decrypt the specific data.

However, a complete database security plan is more elaborate than earlier mentioned above. These include essential experiences like access control, backup, recovery plan, risk management, regular auditing and secure network.

#### **2.2.4 Database Cryptography**

Cryptography is the art of “extreme information security.” It is the sense of securing cryptographic algorithm and database so as to secure it from harmful access (Kevin Kenan, 2006). Encryption is often the first thing that comes to mind when one thinks of data protection. When done properly, encryption is the best way to prevent unauthorized access to data on the database either static or motion. If data leaves the database, encryption is the only protection available. Database encryption can be achieved with third-party applications or natively by the database built-in tools.

Encryption is an essential component of data protection if it is transparent, performs well, and provides built-in key management capabilities (Charles and

Brian, 2008). Cryptographic algorithms can be classified into three categories: symmetric cryptography, asymmetric (or public key) cryptography and cryptographic hashing (Kevin Kenan, 2006): -

- Symmetric Cryptography conventional cryptography or symmetric key cryptography, ensures both sender and receiver use the same key. The famous types available today are: Advanced Encryption Standard (AES) and Data Encryption Standard (DES).
- Public – Key Cryptography known as asymmetric cryptography is a recent invention, where different keys are used for encryption and decryption (private key and public key). When private key is used for encryption, public key must used to decrypt and vice versa.
- Cryptographic Hashing is also known as a message digest, is like fingerprint of some data. It reduces the data to small and unique value.

### **2.2.5 Database -level Encryption**

Database-level encryption allows database owner to secure data as it is written to and read from a database. This type of operation is typically done at the column level within a database table. Database-level encryption protects the data within the DBMS and also protects against a wide range of threats, including storage media robbery, well known storage attacks, database-level attacks, and malicious DBAs. Database-level encryption eliminates all application changes required in the application-level model.

Additionally, careful consideration has to be given to the performance impact of implementing a database encryption solution, particularly if support for accelerated index-search on encrypted data is not used. First, enterprises must adopt an approach to encrypting only sensitive fields. Second, this level of encryption must consider leveraging hardware to increase the level of security and potentially to offload the cryptographic process in order to minimize any performance impact. The

primary vulnerability of this type of encryption is that it does not protect against application-level attacks as the encryption function is strictly implemented within the DBMS. The secure data level encryption for data at rest can be based on block ciphers. The solution provides an effective last line of defense. Selective column-level data item encryption, cryptographically enforced authorization, key management based on hardware or software, secure audit and reporting facility, and enforced separation of duties.

### **2.2.6 SQL Server and Database Encryption Keys (Database Engine)**

Microsoft Corporation (2009c) revealed that, the SQL Server uses encryption keys to help secure data, credentials, and connection information that is stored in a server database. SQL Server has two kinds of keys: *symmetric* and *asymmetric*. Symmetric keys use the same password to encrypt and decrypt data. Asymmetric keys use one password to encrypt data (called the *public* key) and another to decrypt data (called the *private* key).

In SQL Server, encryption keys include a combination of public, private, and symmetric keys that are used to protect sensitive data. The symmetric key is created during SQL Server initialization when you first start the SQL Server instance. The key is used by SQL Server to encrypt sensitive data that is stored in SQL Server. Public and private keys are created by the operating system and they are used to protect the symmetric key. A public and private key pair is created for each SQL Server instance that stores sensitive data in a database.

## **2.3 Database access technologies (DATS)**

In order to understand Database Accessing Technologies (DAT), that can automatic recognize of database table, it is inevitable to review some papers, journals and theses in these areas. Although there are a lot of DATs, the revision was done to

only some of them; those have supported by famous software programs like Oracle, SQL and Java. Therefore, this chapter is in nine sections. A review of Open Database Connectivity (ODBC) comes first. Next, Java Database Connectivity (JDBC) follows. The third section attempts to revise ActiveX data objects (ADO). Fourthly, Online Analysis process (OLAP) was examined. Fifthly, Object linking and embedding (OLE) was revised. Sixthly, Common Object Request Broker Architecture (CORBA) was discussed. Seventhly, ADO.NET was introduced. Eighthly, Microsoft.NET was studied. Finally, Extensible Markup Language (XML) is examined.

### **2.3.1 ODBC**

Open Data Base Connectivity ODBC has many definitions exist in the computing world today. “To the end user, it is an icon in the Microsoft Windows Control Panel. To the application programmer, it is a library containing data access routines. To many others, it is the answer to all database access problems ever imagined (Microsoft Corporation, 2009a).” In other words is a standard database access method developed by the SQL Access group (SAG) in 1992. Moreover, the goal of ODBC is to make it possible to access any data from any application, regardless of which database management system (DBMS) is handling the data (Wikipedia, 2008).

According to works done by Wei Zheng (2000) revealed that “The Standard Open Database Connectivity (ODBC) is Microsoft's strategic interface for accessing data in a heterogeneous environment of relational and non relational database management systems. Using ODBC interface, an application written in SQL can access remote data.”

Paolo Romano (2005) in his PhD research states that “While presenting innovative methodological achievements, all the proposed protocols are designed to be straightforwardly implementable on top of standard, off-the-shelf technologies.



For instance, they can operate in environments with database servers behaving according to standard connection oriented programming interfaces like, e.g., ODBC or JDBC, and exposing the industrial standard XA interface in support of distributed transaction processing.”

Whereas Samuel, S. (2004) mentioned that the “ODBC is designed to be independent of the particular database implementation. By using embedded SQL, through ODBC Application Programming Interface (API), it is possible to access different databases and hide complexity from the application. ODBC is made from different components: standard API, driver manager component (responsible to use right database driver), and other components (for processing SQL statements and returning results to the application). ODBC is able to access different databases located on different platforms. In the client/server environment and dislocated database systems, ODBC includes also any network software necessary for communication with remote database system.”

While B. Frédéricque, S., *et al.* (2007) in their paper concluded that “We currently consider several technologies to provide these functions. The image and the photogrammetric processing will be performed using DVP software components provided by DVPGS. The final results (i.e. model parameters and multi-scale pattern parameters) will be directly stored in a database through ODBC standards.”

As described by Jin-Tsong Hwang (2008) in his paper, “In an ASP web page, the connection between web page and database is based on the connection object of ADO (ActiveX Database Objects). There already is a standard rule for connection between database and web page called ODBC (Open Database Connectivity). This is an Application Programming Interface (API) that allows for data extraction from a database through a unified source. The main task of Connection object is passing the “Connect String” to ODBC for processing. After receiving the “Connect String”, ODBC analyses the string and then communicates with the database by a specific protocol of TCP/UDP 1433. Therefore, the content of “Connect String” needs to include all of the connection information. This information may include type of database, name of the server, name of database, and login name, etc. Once there is a

successful connection, the Connection object will keep all of the connection information in its memory to avoid breaking the connection.”

“MS Access was used at the backend for creating, maintaining and updating of user categories, user tasks, personal space and information resources tables. Additionally, the Java servlets, JavaScript scripting language, the Web HyperText Markup Language (HTML) and Open Database Connectivity (ODBC) were used for developing the various routines of the system as described by N. Meyyappan., *et al.* (2001).”

George, S., *et al.* (2008) described a general outline of how the data were transferred into the MySQL database as “all of the data for the study units and radiocarbon assays were initially entered into Microsoft Access databases. These data were transferred to MySQL using Access’s export functionality and the ODBC database support provided by Access and MySQL.”

Works by Md. Nasir S., *et al.* (2008) have shown that “Based on the proposed method, a profile learning system is developed using Microsoft VC++ connected to Microsoft Access database through an Open Database Connection (ODBC).”

Ikhu-Omoregbe, N. (2008) added that “Physicians access the application from various handheld devices within the hospital. The application user’s interface allows users to access the required medical information. The database server provides data services and data base management system function. The Open Data Base Connectivity (ODBC) is used to support data logic resulting from database queries. The application enables authorized medical personnel access to the addition to WML so as to add interactive functionality to the static WML pages by providing access to an MS Access database through ODBC-JDBC Bridge.”

In a survey developed by Ortolof, H., *et al.* (2005), “At the time of the termination of the EU-project in 2005, a completely new CMS should be available, which enables scientists in the entire field of classical antiquity to manage and

publish their data files via the internet in a rapid and simple manner. For this purpose “multilingual interfaces” are being offered, by means of which the data files can be managed and retrieved in several languages. The CMS is being developed on an open-source basis and will permit, via an ODBC (Open Database Connectivity)-interface, to integrate a great number of already existing data management procedures.”

Jermu, P., and Juho, R. (2008) in their studies summarized that as “The prediction application (Pöllänen, 2001) was implemented as a Microsoft Excel spreadsheet, since the operators were already familiar with the software. The layout of the user interface was designed to mimic the previously used screens. The easy integrability to Excel was the primary choice for selecting the tool. The interface to the production database of the brewery was implemented using Open Database Connectivity (ODBC).” While Cokhan Aydinli (2000) briefed that “Frequently Excel is also used as a database front-end as it offers various data retrieval methods, e.g. reading ASCII or character files, and gets data from any database system supporting the ODBC standard.”

For data integration, Vadim Bichutskiy (2008) alleged that “We integrated computational docking data into the database by connecting the project’s Microsoft SQL Server database to the laboratory’s PostgreSQL database via Open Database Connectivity (ODBC), a standard database from any application. The project database runs on the Windows server while the laboratory database runs on the Linux server.”

Nick, P., and Hussein, D. (2005) stated fact that “All output data may be saved and archived as ASCII files or in an ODBC database format for ease of future reference.”

Maribel, S., and Luís, A. (2008) reported that “The overall set of databases that integrate the Knowledge, Data Repository and the Results Visualisation components were implemented in Access. The data stored in them are available to

the Data Analysis component, or from it, through ODBC (Open Database Connectivity) connections.”

Faisal, I. and Hassan, M. (2003) in their paper mentioned that “The implementation is centered on the two MFC classes to interact with ODBC driver, namely CDatabase and CRecordset. All the class member functions of the two classes throw a CDBException if they fail, which we catch and display appropriate message box to user. This object will interact with the underlying database and connect to it using the installed driver for ODBC.”

Philippe, Th., *et al.* (2005) said that “The size of the wrappers can be computed from these tables. In RDB wrappers, thanks to the use of the ODBC interface, the schema is described by a table, so that each schema construct requires much less LOC than the COBOL approach.”

Boon, L., and Jake, B. (2003) stated that the exam paper records are stored in MS Access which is not a database server and therefore not accessible remotely by an external agent. There are two general approaches for ‘exposing’ the exam paper repositories:

- Migrate the catalogue to a database server such as SQL server or MySQL.
- Use a database linking protocol such as ODBC.

Using the database linking technique allows librarians to continue using MS Access as a front-end for records administration while resolving the problem of remote access. The database server ‘mirrors’ the exam paper records in the MS Access via a ‘link table’ and software (MySQL’s ODBC) bridge; any changes made to the MS Access records would also be reflected in the SQL database.

“It seems that WebMap supports only the IIS server versions 4.0 to 5.0 and utilizes Open Database Connectivity (ODBC) drivers to connect their internal proprietary database format to other DBMS such as Oracle, SQL Server and

Microsoft Access (Intergraph 2006b). MapXtreme allows for database connectivity through ODBC and ADO.NET through backend DBMS such as SQL Server, Oracle Spatial, Microsoft Access, Informix IDS 9.3 and Oracle 9i and 8.1.7. Autodesk's Feature Data Objects (FDO) technology provides the API for accessing and manipulating data from multiple data stores including SDF, SHP, ArcSDE, WFS/WMS, ODBC, PostGIS and MySQL (Erik Harper, 2006)."

L. Cinque, *et al.* (2003) summarized their studies as "The indexing server will initially be realized with MySQL database software, and will also support Oracle, Microsoft SQL Server, Microsoft Access, Sybase, PostgreSQL, Visual Foxpro, Interbase, and generic ODBC. However, the Archive Release module allows users to write their own custom release modules, either to modify the standard release procedure or to release documents into a proprietary back-end or non- ODBC database."

Fredrik Bökman (2001) mentioned that "Wrappers are an important part of the mediator-wrapper architecture of Amos II and PAQS. A wrapper functions as an interface to an external data source, and it is only by means of such wrappers the mediator is able to access external data. The approach has recently been included in the SQL: 1999 standard as SQL/MED (Melton et al 2001, see section 4.4.3). Presently, the Amos II system incorporates wrappers for relational databases (as ODBC data sources) and XML files. It is possible that other wrappers will have to be implemented for access to important data sources. However, this might very well be unnecessary work since there is a very strong trend towards XML formats in life science research."

"The web server seems to be off-line, and people out in the field cannot download the Maefairs application. You spend time talking with the network staff, looking at ODBC drivers, and researching the problem on technet. It seems the new drivers the network staff installed were not compatible with the ODBC configurations. It gets fixed by changing the ODBC parameters on the file server (William J. Hallinan, 2008)."

Works by revealed by Zhiyuan, Sh., and Hai, J. (2008) consider that Clients can connect to each of the server nodes via ordinary connection facilities, e.g., ODBC, and then submit their requests in the form of transactions. A middleware program, whose functionality is to guarantee the consistency of the data copies.

While works by Daan, L., and Erik, M. (1999) have shown that “Databases are ubiquitous in computer science. For instance, a web site is usually a fancy facade in front of a conventional database, which makes the information available in a convenient browsable form. Sometimes, servers are even running directly on a database engine that generates pages from database records on-the-fly. On Unix platforms this is usually done via ODBC, under Windows there are confusingly many possibilities such as ADO, OLE DB, and ODBC.”

“Database access and query ability. Mifflin currently uses the Microsoft Access database. All data is directly accessible via either the Mifflin application, the database itself, or external JDBC or ODBC queries (E. Jeffrey Conklin, 2002).”

Prashant Nair (2003) believed that “When the web server receives this information from the client, it initiates an instance of the search script and executes it. This script, running on the web server, first establishes a connection with the SQL server using an ODBC (Open Database Connectivity) bridge. The HTML pages provide statically or dynamically generated links to access resources in the database on an ODBC bridge or some other connection. An Open Database Connectivity (ODBC) bridge connects the database and the Microsoft Internet Information Services (IIS) web server.”

“In recent years the need to effectively process large amounts of data has become increasingly important. Companies are using databases to store information about for example warehouses, orders, invoices and much more. Because of this, the need to access this data using standard tools has become very important. Since there are many different kinds of databases in use today, a standard method for accessing data was developed by Microsoft, *\_Open database connectivity* (ODBC).

Applications using ODBC as their data accessing method can read and manipulate databases for which there are ODBC drivers (Marcus Eriksson, 1999).”

J-P. Rosen (2008) stated that he had several options for interfacing to MySQL from Ada:

- • Use GNOME with the native interface to MySQL;
- • Use ODBC and GNOME with the ODBC interface;
- • Use ODBC and a direct binding to it.

He rapidly decided to use ODBC, because it would ensure that his application was not dependent on any special feature of MySQL, and would allow him to change the DBMS without changing the application. MySQL provides ODBC drivers for both Windows and GNU/Linux.

“Web applications often use data submitted by a client to construct SQL queries. The communication between the programming language and the database commonly takes place through a call level interface such as JDBC or ODBC. Call level interfaces allow full power of SQL by accepting dynamically generated SQL statements as mentioned by Amit, K. (2007).”

### **2.3.2 JDBC**

The Java Database Connectivity (JDBC) API is the industry standard for database-independent connectivity between the Java programming language and a wide range of databases – SQL databases and other tabular data sources, such as spreadsheets or flat files. The JDBC API provides a call-level API for SQL-based database access.

JDBC technology allows you to use the Java programming language to exploit "Write Once, Run Anywhere" capabilities for applications that require access to enterprise data. With a JDBC technology-enabled driver, you can connect all corporate data even in a heterogeneous environment (Sun Microsystems, Inc., 2009).

Works by Rakesh Dhaval (2005) have claimed that “The communication between the query engine and the database is done via the database interface. The database interface utilized is JDBC. Furthermore, the database connectivity needs to work with Java programming language of choice. We use both the Open Database Connectivity (ODBC) application programming interface (API) along with the Java Database Connectivity (JDBC) application programming interface specifications to connect our Java programs to our database. For our purposes the JDBC-ODBC bridge that comes with the Java SDK works well.”

Shuxin Yuan (2000) named that “Java Bean provides an interoperable way to produce interoperable components. Java Database Connectivity (JDBC) or JDBC-ODBC (Open Database Connectivity) provides a universal means of connection between the internal objects and the external database.”

“The information tier, or data tier, maintains data for the application. The information tier stores data in a relational database management system (RDMS). For this application, we can use an Oracle database that will interact with the web tier through the Java Database Connectivity (JDBC) driver (Mounia Belmamoune, 2003).

Parag, D., *et al* (2004) concluded that “The Data Access Modules retrieve and filter raw data from the data sources and provide refined data to the appropriate Data Services Module. As data sources can be of various types, our tool supports databases as well as various probabilistic modelers. When databases are used as a data source, we use JDBC-ODBC to access data sources. Java Database Connectivity (JDBC) – Open Database Connectivity (ODBC) provides a means to access databases through programs written in the Java high-level programming language.



Through JDBC-ODBC, the use of a Java-based EIS tier allows efficient communication between the server-component and the database system.”

“In this prototype, extensive use was made of JDBC to query the databases, construct the Local Database, and handle the table objects. This was convenient because Informix supports JDBC driver. The object support within Java and JDBC also simplified many of the internal structures. It is entirely consistent with this architecture to use other methods, including ODBC and CORBA as described by B. Fu, S., *et al.* (2001).”

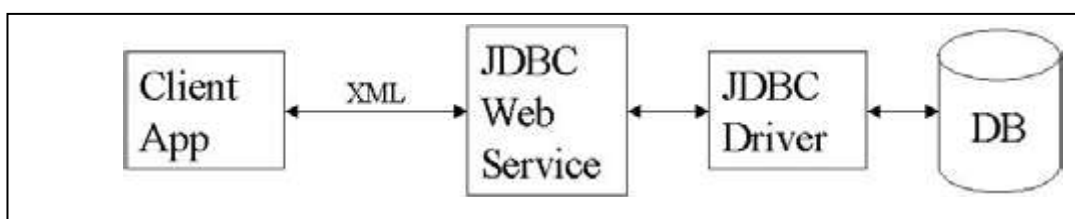
According to Simon Kainz (2008), “The search module developed in this thesis uses Spring's JDBC framework for database communication. Based on the concept of JDBC templates, reliable and secure type-safe database queries are created based on predefined templates. By using Spring JDBC-templates querying databases is as simple as specifying a query statement and providing a result-to-data object mapping for each result. All underlying steps necessary for database access are handled transparently by the Spring JDBC package.”

Awais, R., and Ruzanna, Ch. (2003) noted that “there are a number of JDBC drivers available. We have chosen to base the implementation of the Database Access aspect on the basic Sun Microsystems JDBCODBC Bridge Driver which, to the best of our knowledge, offers the lowest common denominator in terms of supported functionality.”

Talha Oktay and Murat Unal (2000) thought that “Oracle 8i is the latest version of the Oracle Corporation’s DBMS can be solution to this problem. With Oracle 8i’s Java-enabling components-Object Request Broker (ORB), Java Virtual Machine(JVM), and embedded JDBC Driver- Turkish Navy have a wealth of technologies at its disposal. Turkish Navy has a choice of several programming models—PL/SQL, JDBC, SQLJ, CORBA, and EJB; and a choice of protocols—Net8 and CORBA-IIOP.”

Erdogan, D., *et al.* (2004) found that the relational databases can be accessed by general-purpose Java applications using standard JDBC API. This requires applications to use a JDBC driver specifically developed for JDBC access to the database system that needs to be accessed.

Furthermore, they implemented a database web service called JDBC Web Service (JDBC-WS). JDBC-WS will provide a standard JDBC-like interface that can be used by any Web Services client. See figure (2.2).



**Figure (2.2):** Web Services-based access to relational databases

“In the Tele-Electrogastrgraphy (EGG) application, Java client applet can start plotting EGG data by invoking the relevant remote object method on the web server that can obtain access to a database using Java Database Connectivity (JDBC) Data Access Application Programming Interface (API). This interface is contained in a library that is known as an Open Database Connectivity Bridge (ODBC), which implements JDBC according to the ODBC standard C API. Software developers can utilize JDBC to send SQL statements to a database and retrieve the results of a query. JDBC-ODBC is a JDBC driver that provides access to any ODBC database including Microsoft SQL or Microsoft Access from any Java platform as stated by Simon, Zh., *et al.* (2002).”

“Java Servlets support the Java DataBase Connectivity (JDBC) API to access databases that support the JDBC API, too. SQL Server supports the Open DataBase Connectivity (ODBC) API that can be linked to the JDBC API as concluded by Ulrich, U., and Stephanie, T. (2001).”

Gary Yeung, (1999) in his thesis mentioned that “The choice of Microsoft SQL Server 2000 was based on its availability for this project. The basic requirements for the data source are that it must be able to execute SQL statements and it must be a JDBC or an ODBC data source. This is because the multi-agent IIVM system is created in Java and it can only communicate with a JDBC data source or an ODBC data source through the ODBC-JDBC bridge. JDBC and ODBC will be discussed later in this chapter.”

Gustav Boström. (2003) in his research found that “Lumbago was constructed using standard Java. The graphical user interface was built using Swing and the business logic was implemented in plain old Java classes that access a HSQLDB database using direct JDBC.

Ke Wei (2006) recommended that “JDBC (Java Database Connectivity) and Ado.net (ActiveX Data Objects) are API that defines how a client application may access a relational database. It provides methods to query and update data in a relational database.”

### **2.3.3 ADO**

ADO is part of the MS Universal Data Access (UDA) strategy to make all data accessible using the same set of procedures. This applies to data held in formats as diverse as text, XML or proprietary DBMS formats. ADO uses a technology called OLEDB which was specifically designed by MS to replace ODBC and is based on the company's Component Object Model (COM). While, Khalid Eldrandaly *et al.* (2005) in their research revealed that, two different COM-complaint software packages are used: ESRI-ArcGIS 8.2 and Microsoft-Excel 2002. Visual Basic for Application (VBA) was used to develop an Excel application that implements the AHP technique. Microsoft ActiveX Data Object (ADO) was used within the AHP Excel application to read required information from the geodatabase. ADO was

implemented using a set of COM-based interfaces that provide applications with uniform access to data stored in diverse information sources.

The server application uses version 2.7 of the Microsoft ActiveX Data Objects (ADO) in Visual Basic 6.0 to enable connectivity with the data source, Microsoft Access database. “With every ADO, a structured query language (SQL) statement is used in the server application to access and manipulate the data stored in the database (Agustinus, 2005).”

José A. B., *et al.* (1998) concluded that “While OLE DB is a powerful API for manipulating data, most application developers do not need the fine-grained control that OLE DB offers. Most developers are not interested in managing memory resources, manually aggregating components, and other low-level operations. ActiveX Data Objects (ADO) is an application-level data access object model that allows easy access to any OLE DB data source from languages such as Visual Basic, C++, Java, VBScript, and JavaScript.”

“The system works using Active Server Pages (ASP) with an ADO/ODBC connection to the Oracle database. ODBC (Open Database Connectivity) is a C-level application programming interface (API) for SQL data. While developers can code directly to the ODBC API it can be fairly difficult and involve a lot of code. Because of this, Active Data Objects (ADO) are used as the interface to ODBC. ADO is an application-level programming as described by S.C. Rennie, *et al.* (2000).”

Ding, L., *et al.* (2007) found that use ADO database accessing technique to accomplish data storage. ADO enables application to access not only relational database but also nonrelational database and provides consistent, high-performance access to data, whether we are creating a front-end database client or middle-tier business objects using an application, tool, language, or even an Internet browser.

Works by Steven J. S., *et al.* (2004) have shown that “The purpose of the database in this simulator is to supply information to the simulation engine. Data are

stored in a Microsoft Access 2000 database, and the simulator uses ADO to connect to the database. This allows the user to override the development platform and substitute the current implementation with a Microsoft SQL Server, an Oracle database, or any ODBC database.”

#### 2.3.4 OLAP

Online Analytical Processing (OLAP) tools have emerged to address the limitations of traditional databases and spreadsheet applications. Unlike traditional databases, OLAP tools support a range of complex analytical queries; unlike spreadsheet applications, they can also handle massive datasets. Moreover, OLAP tools can process user queries online.

Owen, K., and Daniel, L. (2003) concluded that “On-line Analytical Processing (OLAP) is a database acceleration technique used for deductive analysis. The main objective of OLAP is to have constant-time or near constant-time answers for many typical queries. Using OLAP, however, the computation can typically be done on-line.”

“Data warehouse systems are implemented using a set of commercial components (e.g. extraction tools, database systems, metadata management systems, OLAP tools, report generators) that are configured according to the needs of the project. In order to study the principal issues involved in automatic data warehouse generation we chose two typical real world OLAP products as target systems as described by Karl, H., *et al.* (2000).”

Thomas, P., and Toby, J. (2002) found that “data warehouses have been engineered to answer queries of aggregation with “group by” expressions efficiently. The goal of on-line analytical processing (OLAP) is to quickly answer queries from large amounts of data residing in a data warehouse. To improve the quickness of response to queries, pre-aggregation is a useful OLAP strategy.”

Works by Fangyan, R., *et al.* (2003) have shown that “Data warehouse is a subject-oriented, integrated, time-variant, and non-volatile collection of data in support of management’s decision making process. The purpose of data warehouse is to support OLAP, which is especially designed for knowledge workers (executives, managers, analysts) to systematically organize, understand, and use their data to make strategic decisions.”

Moreover, Surajit, Ch., and Umeshwar, D. (1997) announced that “OLTP applications typically automate clerical data processing tasks such as order entry and banking transactions that are the bread-and-butter day-to-day operations of an organization. Typical OLAP operations include rollup (increasing the level of aggregation) and drill-down (decreasing the level of aggregation or increasing detail).”

Data mining and OLAP are complementary as they are both analytical tools. For example, in the customer dimension of sales cube, there are lots of members, so it is very difficult to find customers’ buying patterns. Data mining techniques can be applied to analyze this dimension to find out what are the clusters among the customers based on customer member properties and measures. SQL Server Analysis Services has advanced features that bridge data mining and OLAP (Zhaohui, T., *et al.*, 2005).

Denis and Marie-Chantal (2008) found that once the systems were integrated within their data warehouse environment, the data was made available to users either in the form of reports using various data formats (i.e. web report, MS-Excel, etc.) or OLAP analysis.

### **2.3.5 OLE**

OLE stands for Object Linking and Embedding. It is a technology for transferring and sharing information among applications. Different applications

expose their objects that are related to the kind of data the application works with. Automation client is an application that exposed objects belonging to another application. For instance VB program acts as an Automation client. An Automation server is an application that exposes programmable objects to other applications. For example Excel can act as an Automation server. Excel exposes Automation objects. These objects have properties and methods as their external interface (S.S. Ahmed, 2002).

“The world of Windows programming is moving toward a division of labor in which C++ programmers will produce reusable modules that will be integrated into applications by Visual Basic (VB) programmers via tools like OLE Automation, Using OLE Automation. It is possible for one application to control another (Raymond P., and Kirsch, 1996).”

Zhaohui, T., *et al.* (2005) introduced that “A data mining model is a new concept introduced in OLE DB for DM. With the help of the OLE DB for DM Specification, any data mining algorithms can be accessed through OLE automation, which can be easily embedded into any consumer applications. Moreover, Microsoft initiated the OLE DB for Data Mining (DM) Specification with more than a dozen independent software vendors (ISVs) in business intelligence.

JOS6A. Blakeley (1996) stated that OLE DB enables the development of an application that will access both information sources and make it easy to access data stored in diverse DBMS and non-DBMS information sources. DBMS sources may include mainframe databases (e.g., IMS, DB2), server databases (e.g., Oracle, SQL Server), or desktop databases (e.g., Access, Paradox, Fox). Non-DBMS sources may include information stored in file systems (e.g., Windows NT, Unix), indexed-sequential files, email, spreadsheets, project management tools, and many other sources.

Campbell, D.G. (1996) in his paper abstracted that “These economic benefits have been achieved through producing large numbers of systems from standardized components. The OLE Transaction Architecture defines a set of distributed

Component Object Model (COM) interfaces that can be used to coordinate distributed transactions using standard two-phase commit protocols. One of the goals of this architecture is to support the development of commodity enterprise computing software. A crisply defined and unambiguous transaction coordination architecture is a foundation for advanced applications requiring distributed transaction support.”

“Works by Anwar, M.R., *et al.* (2004) depends on using OPC (OLE for process control) as an Interface Layer between industrial devices and HMI. Moreover a Database is also required to store process values for components.”

“The possibility of data transfer between control and diagnostic systems is provided by the using of OLE for Process Control (OPC) standard, where OLE is Object Linking and Embedding. OPC is a client and server technology. The first uses the data and the second provides it (Natalia Sekacheva, 2007).”

### **2.3.6 CORBA**

There has been much works done in the field of CORBA user interface. Desmond, Ch., *et al.* (2001) in their research claimed that “CORBA specifies a system that provides interoperability between objects in a heterogeneous, distributed environment and in a way transparent to the programmer. Its design is based on the Object Management Group (OMG) Object Model. This Model defines common object semantics for specifying the externally visible characteristics of objects in a standard and implementation independent way. In this model clients request services from objects (which are implemented as servers) through a well-defined interface. OMG also specifies a set of Object Services. These are a collection of services (interfaces and objects) that support basic functions for using and implementing objects.”

Mária, T., *et al.* (2003) announce when designing a Universal personal computing, they selected CORBA as a natural platform-independent framework for



UPC, and Java for the implementation of client side objects and agents of UPC. However they found that “CORBA provides transparent access and location independence with the use of object references; i.e., the client does not need to know the address or the actual location of a requested object. Instead, it is the ORB’s responsibility to find or create the appropriate object based on the object reference submitted by the client. The object reference is a unique identifier of an object instance generated at its creation. Implementation independent access to objects for any client is achieved by the standardization of the CORBA Interface Definition Language (IDL). All CORBA object interfaces should be specified in IDL. From an IDL specification, invocations can be derived at run-time. The standard includes also language mapping between IDL and programming languages (C, C++, SmallTalk, Java, etc.). This way CORBA enhances the underlying technology to create a homogeneous DPE as described by.”

There are two ways of object invocations at the client side:

1. If the object interface is known at the client’s compilation time, the invocation can be built into the client code in a static way with IDL stubs;
2. Otherwise, CORBA supplies the Dynamic Invocation Interface to let the client discover methods to be used at run time.

“Heterogeneity is natural in distributed object computing systems. CORBA provides inter-operability in such heterogeneous environments. Additional overheads incurred in the CORBA compliant ORB deteriorate system performance. Achieving high performance in CORBA-based systems by using innovative client-middleware-agent architecture and by exploiting limited heterogeneity in systems are discussed in the literature (Weili Tao, Shikharesh Majumdar, 2003).”

“Works by Alessio, B., *et al.* (2002) have shown that in developing distributed applications, a simple solution for prompt integration of modules communicating by means of either CORBA or RMI is the introduction of a gateway module. This solution determines an additional communication overhead.”

Matteo, P., *et al.* (2003) noticed that Instrumentation Maintenance System (IMS) does not communicate directly with the field devices: and claimed that “One possible solution in order to overcome this situation is to use the high level abstract interface provided by CORBA to define an open environment in which different applications can coexist and share information. In TC, IDL interfaces are modeled by the meta-class Interface. Thus, a CORBA application object implementing a CORBA IDL interface is modeled by an Application Object class inheriting from an Interface class modeling the latter. In this way different Application Object classes might be designed to provide different semantics to the same Interface class, according to the definition of CORBA IDL interface as introduced by.”

While works by E-Kai, Sh., *et al.* (2000) claimed that “The CORBA trading service allows retrieval of object references with respect to some of the characteristics or quality of service (QoS) that objects can provide. An agent that provides the trading service may need to choose different server objects at different points in time for providing the same service. For example, Server A may be suitable for providing a certain QoS at the current point in time but Server B may become the appropriate choice at a future point in time when Server A is running at a higher load. Instead of using the same handle in subsequent requests to the same server, routing every request with the help of a trading agent is important in such a situation.”

Darrell, B., *et al.* (2001) thinks that when integrating Asynchronous Method Invocation (AMI) model into the CORBA specification. AMI improved the scalability of clients by removing the restrictions associated with Synchronous Method Invocations (SMI). Server request handling remained synchronous, however, which minimized the benefits of AMI for middle-tier servers, such as firewall gateways and front-end database servers.

However, works by Liang, G., *et al.* (2000) have show that “Beowulf applications have been primarily in the scientific computing domain. Adding Java/CORBA technology allows for many additional application areas in the area of client/server computing such as database services.”

C. Liebig *et al.* (2000) summarized that the need for supporting information dissemination applications by integrating relational databases in a fully distributed Object-Web system. They had shown that this can easily be done in a CORBA framework through the use of a publish/subscribe Persistent State Service.

Software Environment of the prototype that they developed uses CORBA, VisiBroker for Java, databases (Oracle, mSQL, DB2, and ObjectStore). The JDBC Bridge is used to connect the CORBA server objects to their relational databases. In this case, the CORBA objects are implemented in Java. The user interface is implemented as a Java applet that communicates with the CORBA objects as described by (Athman, B. *et al.*, 1999).

Works by Jianqiang, H., *et al.* (2004 ) have revealed as “Obviously enterprises will still use a variety of different middleware technologies and Web Services can bridge the existing boundaries. The major contributions of the paper are as follows: (1) present static and dynamic models to wrap CORBA objects as Web Services; (2) solve three key technologies in CORBA Web Services: scalable architecture, SOAP/IOP protocol data type mapping and unified service providing infrastructure. We have implemented the prototype system Star Web Service and our test results illustrate the effectiveness of the proposed models and architecture.”

Oliver, K., and Alex, J. (2000) integrate CORBA into the WOS (Web Operating System). They chose CORBA because it is the de-facto standard for distributed object-oriented systems and therefore allows the WOS to be used with real world applications.

### **2.3.7 ADO.NET**

ADO.NET, ActiveX Data Object, is a component technology in .NET used when access to data sources is needed or manipulation of data in a data source is

wanted. The data source can be Microsoft SQL Server as well as data sources exposed via OLE DB and XML. ADO.NET was designed to meet goals as disconnected data architecture (the DataSet), common data representation with the ability to combine data from multiple and varied data sources, and maybe most important; to apply a liberal use of XML infrastructure (Julia Norman, 2002).

Vorgelegt von (2006) has shown that the connections to remote databases are established using the ADO.NET framework based on the ActiveX Data Object (ADO) technology. Furthermore, Visual Basic and C# support remote procedure calls.

William Stott (2007) stated that “The Database and SQLUtils classes facilitate this process using the ADO.NET SQL Server Data Provider. Alternative classes could be developed at a later date to support other Data Providers like OLE DB or ODBC, if required.”

Václav Skala and Tom Philip (2004) recommended that “The database is implemented based on the Microsoft SQL Server 2000 and is available for the applications over ADO.Net. The data transfer between the database and the LIVE-Fab program modules occurs through XM.”

“Microsoft .NET framework is a proprietary technology that runs only on Microsoft Windows platforms. Microsoft defines the .NET Framework as “as an integral Windows component that enables building and running the next generation of software applications and Web Services. It includes technologies for Web Services and Web applications (ASP.NET), data access (ADO.NET), smart client applications (Windows Forms) and many others (Tom Philip, 2004).” In another way, we can always use ActiveX Data Objects (ADO) under Visual Basic.NET, which is currently called as ADO.NET. ADO.NET is important for us in working on database connectivity as described by Fong Yit Meng *et al.* (2005).”

Works by Swapna Kodali (2007) have found that “In e-commerce applications it is very typical for the Web server to contact the database to get information as needed. ASP.NET uses a technology called ActiveX Data Objects.NET (ADO.NET) to connect to the database.”

“The resulting command can then be executed to produce the relevant results. In addition to working against various versions of SQL Server, the Entity Framework is being integrated with various third-party ADO.NET providers for DB2, Oracle, and MySQL as mentioned by (Atul Adya, *et al*, 2007).”

### 2.3.8 XML

The Extensible Markup Language (XML) is a general-purpose specification for creating custom markup languages. It is classified as an extensible language, because it allows the user to define the mark-up elements. XML's purpose is to aid information systems in sharing structured data, especially via the Internet.

XML's set of tools help developers in creating web pages but its usefulness goes well beyond that. XML, in combination with other standards, makes it possible to define the content of a document separately from its formatting, making it easy to reuse that content in other applications or for other presentation environments. Most importantly, XML provides a basic syntax that can be used to share information between different kinds of computers, different applications, and different organizations without needing to pass through many layers of conversion.

XML began as a simplified subset of the Standard Generalized Markup Language (SGML), meant to be readable by people via semantic constraints; application languages can be implemented in XML. These include XHTML, RSS, MathML, GraphML, Scalable Vector Graphics, MusicXML, and others. Moreover, XML is sometimes used as the specification language for such application languages (Wikipedia, 2009).

Short for Extensible Markup Language, a specification developed by the W3C. XML is a pared-down version of SGML, designed especially for Web documents. It allows designers to create their own customized tags, enabling the definition, transmission, validation, and interpretation of data between applications and between organizations (Jupitermedia Corporation, 2009).

Andrey, B. and Yannis, P. (2004) revealed that “the approach towards building XML DBMS’s is based on leveraging an underlying RDBMS for storing and querying the XML data. This approach allows the XML database to take advantage of mature relational technology, which provides reliability, scalability, high performance indices, concurrency control and other advanced functionality.”

“XML has become the de facto standard for information/data representation and exchange on the Internet and elsewhere. It has been a wide consensus that XML documents/data should obtain the same type of management functionalities as conventional data received from RDBMSs, and the database community is well underway towards this destination. In recent years, many storage schemes for XML data have been proposed, e.g., mapping XML data to relational or object-relational models, using special-purpose databases such as semi structured databases, or developing native XML databases. A key issue that faces every XML data management system is the optimization of XML queries as described by Dunren Che , *et al.* (2006).”

Zachary G. I. *et al.* (2002) conclude that “The emergence of XML as a common data format, as well as the support for simple web-based query capabilities provided by related XML standards, has suddenly made data integration practical in many more cases. As a result, XML has become the standard format for data dissemination, exchange, and integration. Nearly every data management-related application now supports the import and export of XML, and standard XML Schemas and DTDs are being developed within and among enterprises to facilitate data sharing (instances of these are published at the BizTalk and OASIS web sites1).”

Felix Weigel, *et al.* (2005) introduced (Content-Aware Data Guide) RCADG as an embedding of native XML indexing techniques into a relational database system (RDBS). By integrating native XML indexing techniques even deeper into the core of an RDBS, they expected further improvements. In particular, they outlined how the RCADG can provide XML-specific statistics to the relational query planner and optimizer to make them aware of the hierarchical structure of the data.

According to works done by Airi, S., and Frank W. (2008); in an XML database they should be able to express queries in terms of all data in the database, including entities, URIs, tags, comments, processing instructions, schemas and other metadata.

Samuel and Philippe (2008) have shown that the first storage option is to load XML documents within the rows and columns of a standard relational database with no special extension managing XML. The mapping between the XML fields and the tables/rows is no easy business as XML documents have an unstructured content with a random shape while a RBMS manages rows and columns.

For Data Exchange and Workflow Shankar Pal. *et al.* (2005) mentioned that “XML allows a platform-independent way of exchanging data among applications. The data can be modeled as messages with XML markup. Instead of constantly shredding and generating XML messages, it is prudent to store messages in XML format.”

Also, they think that “XML-based standards are emerging for different, vertical domains, such as for financial and geo-spatial data. These standards describe the structure of the data, based on which instance data can be queried and updated.

Anastasios, I. (2004) claimed that “The Service Creation component is responsible for the design and specification of services, which are to be deployed in the PoLoS Kernel. Service logic is specified using a new language, SCL (Service Control Language) developed for supporting location-based services. SCL is an

XML-based language, built to facilitate implementations with simple functionality but also flexible enough to have the capabilities of a general-purpose language.”

“Today DB2 UDB XML Extender not only serves as a repository for both XML documents and their Document Type Definitions (DTDs), but also provides data management functionalities such as data integrity, security, recoverability and manageability. User has the option to store the entire document as an XML user-defined column or to decompose the document into multiple tables and columns. Fast search via indices is provided for both XML elements and attributes. Section search can be done against the content of the document as said by (Cheng, J. and Xu, J., 2000).”

Muralidhar, K. *et al* (2005) concluded that “XML has become the de facto standard for data storage, processing and interchange between applications. Having an XML datatype in a SQL system enables applications to use XML as the first-class built-in datatype and use it as datatype for columns of tables and views, and parameters and return values of user defined SQL functions, stored procedures, and triggers.”

Works by Dimitris, A., and John, S. (2005) presented recent advances in using XML technologies for network management. Moreover, it introduced a framework for structuring and executing XML management applications. This framework exploits XML technologies in order to specify XML APIs for network devices, but also composite networks.

Jung, K., and Hyunchul, K. (2004) found since emergence of XML as a standard for data exchange on the Web, today’s Web applications are to retrieve information from the remote XML source across the network. Cache-answerability for XML queries is thus crucial for efficient support of XML database-backed Web applications.



### 2.3.9 .NET

Microsoft .NET is an execution environment for Windows programs. It includes two main components: the common language runtime and the .NET base class library. The runtime and core parts of the base class library are specified as an open standard. This standard is called the common language infrastructure, and is published as the ECMA-335 (European Computer Manufacturers Association). In .NET, code runs in Common Language Runtime environment and has automatic memory management. Moreover, .NET provides built-in services for proxies, marshalling, network streams and remoting (Bo Feng and Gabriel Wainer, 2008).

According to Don Syme (2006), “Given this heterogeneity it is somewhat inevitable that people turn toward extensible compilation and intentional meta-programming to bring a degree of uniformity to programming. For example, mainstream languages are now using small meta-programs to specify SQL queries, as in the Microsoft LINQ extensions for Visual Basic 9 and C# 3.0, an initiative that forms part of the background to this paper. One of the motivations of LINQ is to reuse existing infrastructure (e.g. rich editing environments) in the context of embedded domain-specific languages. The end result is that environments like .NET are moving to incorporate a layer of intentional meta-programming, including components to translate meta-programs to languages such as SQL.”

Also Viktor Geller and Christelle Scharff (2005) mentioned that “.NET is a platform that allows secure development and cooperation of programs that are written in different .NET compliant languages; cross-language calls and cross-language inheritance are permitted. There are more than 30 .NET compliant-languages including C#, J#, VB .NET, and Microsoft Visual C++, but also the .NET versions of functional languages such as SML, Scheme, Haskell, imperative languages such as TMT Pascal and Fortran, and object-oriented languages such as Eiffel and Smalltalk.”

While in the survey by Yanhao Zhu, *et al.* (2005), “In the user program, the class has to be defined somewhere before it can be used. This means that the types of

objects which the user wants to deal with have to be declared first in the program. In C#, VB.NET, VC++.NET and other strong typed .NET programming languages; the compilers will enforce this requirement without any exception. The classes of the stored objects, which will be used in the program, have to be defined.”

## **2.4 Summary**

A review of Database and Database Access Technologies clarified the close interaction between them. This is manifested in the establishment of their link to the database and creates a suitable interface environment. However, there are some standard rules of connection between database and User Interface, such as ODBC, JDBC, ADO, ADO.NET, OLAP, XML, OLE and .NET etc. These are Application Programming Interfaces (APIs) that allow for data extraction from a database through a unified source.

## **CHAPTER 3**

### **RESEARCH METHODOLOGY**

#### **3.1 Introduction**

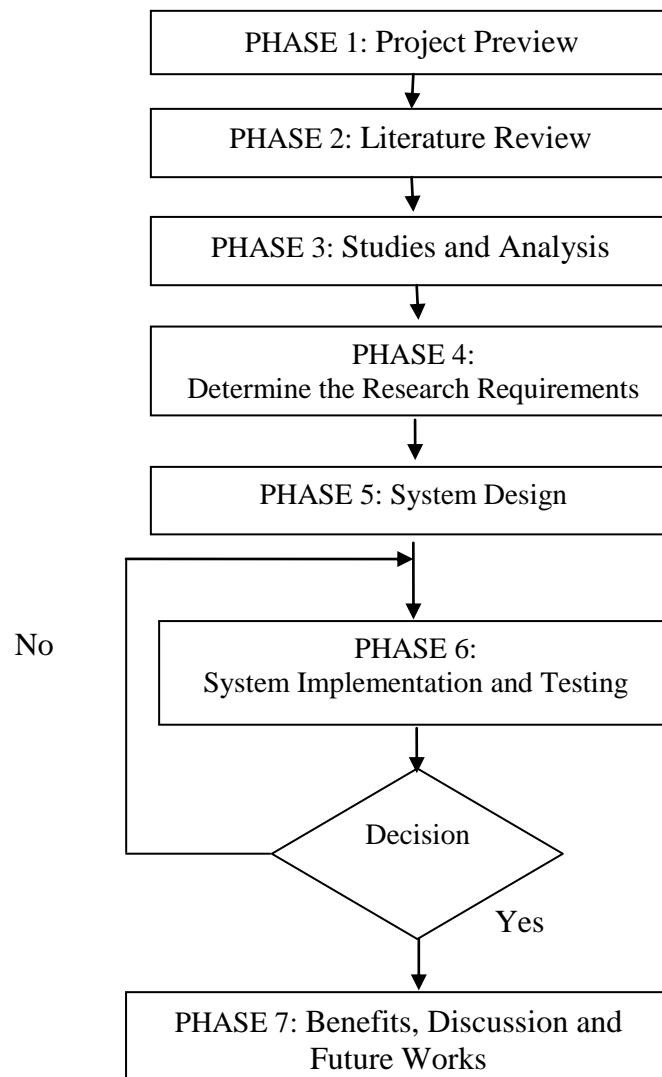
In this chapter the methodology of the project will be discussed. It will reflect the aims and objectives in Chapter 1. However, the methodology of the proposed approach will be described in principle, well-managed and consistent manner. The chapter begins with the research development phases (Section 3.2), in which the project phases are explained. Section 3.3 is the summary.

Methodology represents a set of practical ideas and proven practices for a given area of activity, techniques and methods used in a system during planning, analyzing, design, development or management of computer application.

#### **3.2 The Research Development Phases**

This research has aimed to enhance implementation of database column encryption through Database Access Technology (DATs). Thus, defining and selecting a suitable technology need some knowledge and investigation of existing some DATs. Therefore, this project is conducted according to the workflow process as illustrated in Figure 3.1. The research development is conducted in a phased approach that can help in organizing the required job in a systematic manner. The

research development is divided into seven phases. The following sub-sections describe these phases.



**Figure (3.1):** Research Flow Chart

### 3.2.1 Project Preview

Problem sharpness and formulation is the first stage held by the thesis. In that section the database column encryption/decryption through Database Access Technologies were considered.

### **3.2.2 Literature Review**

Literature review is defined as a body of text that aims to review the critical points of current knowledge on a particular topic. However, the goals are to bring up to date information about the others done in the same field. For this thesis, the literature review introduced database, database Cryptography, Database -level Encryption.

### **3.2.3 Studies and Analysis**

The studies and analysis, and literature review have aimed to provide an overview and recent progress in DATs automating service to detect database; and how to facilitate column encryption. These approaches have classified into some categories. But it is impossible to claim that this classification is very exhaustive. In each category, the introduction, features/capabilities, advantages, disadvantage and column encryption support are stated in details.

### **3.2.4 Determine the research requirements**

As this project is concerned about the enhancement implementation of encryption/decryption of database through DATs, building a system based on this standard becomes necessary. This standard requires two main entities: hardware and software.

Based on these requirements, it was found that Windows XP/Vista operating systems and SQL Server 2005 installed on client desktops can comply with these standard requirements. For more details see section 5.4 system requirements.

Building the system in client/server environment requires the following:

- i. Hardware requirement specification

- ii. Software requirement specification

### **3.2.4.1 Software Development Tools**

#### **A. Microsoft Visual Studio.Net**

Microsoft Visual studio VB.Net was chosen as programming language to develop the necessary GUIs, because of its easy features (drag and drop), Microsoft Windows based applications, compatible with SQL Server 2005 and the main feature is an Object Oriented Language, and it supports CLR, the new development feature, which integrated .Net framework with MS SQL server 2005.

#### **B. Database Management System (DBMS)**

Microsoft SQL Server 2005 used as a backend too with a powerful mechanism (Encryption Mechanism) created using the .Net CLR (Common Language Runtime) which consists sets of Stored Procedures (SProc) and User Defined Functions (UDF) used for transparent encryption and decryption.

Works by Mitch Ruebush (2004) compared SQL Server and Oracle 10g and said that “As a result, both Oracle and Microsoft have integrated their database offerings (Oracle 10g and SQL Server 2005 respectively) with Visual Studio and the .Net platform. Integration with Visual Studio and the .Net CLR represents one of the biggest advancements in developer productivity for database application development in the last decade.”

- **Microsoft SQL Server 2005:** Undoubtedly the most significant new feature in the SQL Server 2005 release is the integration of the Microsoft .Net Framework. The capability of the SQL Server is extended in several important ways especially the integration with the .Net CLR. The integration of the .Net with SQL Server 2005 enables the programmers using any .Net language supported CLR like VB.Net, C# or C++ to develop SProc (Stored

Procedure), User Defined Functions, aggregations, triggers, and user-defined types.

- **Stored Procedure (SProc):** SProc used to be created using Transact-SQL (T-SQL) which is always difficult to write a code either using T-SQL or extended SProc/Function using COM. So the CLR, SQL Server 2005 allows us to deploy C++, VB.Net or C# code that is used within the SQL Server process. This means that if you need complex procedural code, you can write it as managed code.
- **User Defined Function (UDF):** A user-defined function is a database object introduced in SQL Server 2000. It comes in two flavors: scalar-valued UDFs and table valued UDFs. A scalar-valued function returns a single value for each function call, while a table valued UDF returns a table record set that can be joined with the others tables/result sets.

A new feature in SQL Server 2005 is the apply operator, which allows the database developer to invoke a user-defined function. Creating .Net base UDFs is another new feature that's enabled by the integration on the .Net CLR.

- **User Defined Triggers (UDT):**In addition to stored procedures and User Defined Functions, the new integration capabilities in SQL Server 2005 also provide the ability to create .Net User Defined Triggers which created from Secure Column Application during first time of encryption a sensitive table cause the execution of the stored procedure or a user defined function at run time when a request is performed from the database server and they already registered in the SQL Server database during the initialization of the Secure Column Application.

### C. Documentation Tools

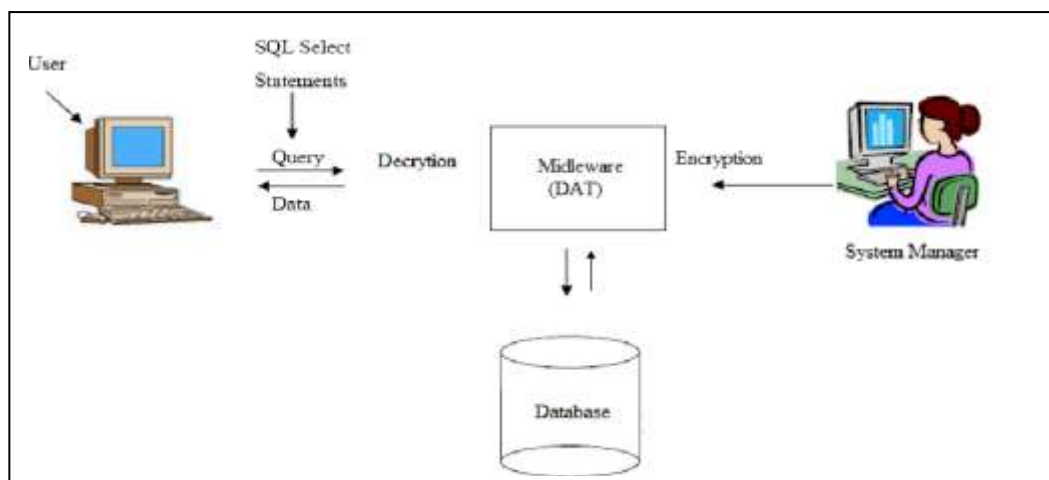
Microsoft Word 2007 and Notepad are used to document project process

## D. Project Schedule

Refer to Appendix E for the project Ghant Chart that has the details of the carrying tasks and time allocation.

### 3.2.5 System Design

Throughout the literature review that had done in Chapter 2; especially the different capabilities of DATs such as supporting internet, network and database encryption. The scheme design can be shown in Figure 3.2. This figure shows the overall structure of the scheme.



**Figure (3.2):** The overall Project Layout

#### 3.2.5.1 Database design

Database design produces a detailed data model of a database. Secure Column Application targets Microsoft SQL Server 2005 as a development and deployment environment. The Secure Column Application secures the above mentioned relational database management system RDBMS from unauthorized access and makes sure of the integrity, confidentiality and availability of the data to the authorized users only.



Secure Column Application's data storage is designed in a schema named Scolumn, which is accessible only to the authorized users. In the next section (5.3), detail information on Secure Column Application's database design such as the tables used by the Scolumn schema will be provided:

### **3.2.6 System Implementation and Testing**

In this phase, an implementation based on the previous schema design is established. This is a lab-based project environment. The windows Vista, SQL Server 2005 Express Edition and VB.Net are installed on standalone unit.

Also this phase tests the implemented system to see whether it meets the project requirements. The test plan emphasizes on the plan for testing the Secure Column Application to realize the following objectives:

- To develop test documents.
- To develop a cryptographic database security mechanism for SQL Server 2005;
- To develop the initialization and authentication feature for the Secure Column Application;

The plan will address System Test of the Secure Column Application, as it will be performed by the Quality Assurance (QA) group of the Scolumn team. Unit testing is the responsibility of development engineers.

The types of testing that will be addressed during System Test are Functional Testing, Integration Testing, User Interface (UI) Testing and Performance Testing. The major use cases to be utilized for developing the test requirements are the Login, Encrypt Data, Decrypt Data, Generate Key, Manage User Info use cases. Encrypt Data, Decrypt Data and Generate Key will be the main focus for the integration tests.

### **3.2.7 Benefits, Discussion and Future Works**

The final chapter of this thesis is mainly discusses the implemented system, which could lead to future works necessary to enhance the effective implementation. For instance an implementation modules, lessons learnt, expected organizational benefits, extends suggestions for the further research into the future in the area of level security schemes for database including cryptography and database security in common.

### **3.3 Summary**

This chapter discussed the methodology used in implementing the project. It consists of five stages as follows: determination of the research requirement specification, system design, system implementation, system testing and report writing. Each of these phases plays an important role in accomplishing this project. The following chapter reviews DATs and their analysis (Chapter 4).

## CHAPTER 4

### STUDIES AND ANALYSIS'S

#### 4.1 Introduction

The objective of this chapter is to study and analyze possible technologies use for connecting to central database and to implement a remote client connection in practice. In this analytical section, connection technologies, such as ODBC, JDBC, CORBA, OLE, OLAP, ADO, XML, ADO.NET and Microsoft.NET are analyzed. In addition, the theoretic part gives comparison of technologies in different aspects like support internet, network and database.

#### 4.2 The Java Database Connectivity (JDBC)

A JDBC driver is a software that enabling a Java application to access a database. However, to connect with individual databases, JDBC (the Java Database Connectivity API) requires drivers for each database. The JDBC driver allows the connection to the database and implements the procedure for exchanging the query and result between client and database.

##### 4.2.1 JDBC API Overview

The JDBC API makes it possible to do three things:

- Establish a connection with a database or access any tabular data source
- Send SQL statements
- Process the results

#### **4.2.2 JDBC Architecture**

The idea behind JDBC is like to Microsoft's Open Database Connectivity (ODBC). However, all of them depend on the X/Open standard for database connectivity. Applications written by using the JDBC API should communicate with a JDBC driver manager that uses the current driver loaded.

The JDBC API offer two basic sets of interfaces: the first is the JDBC API for application writers, and the second is the lower-level JDBC driver API for driver writers.

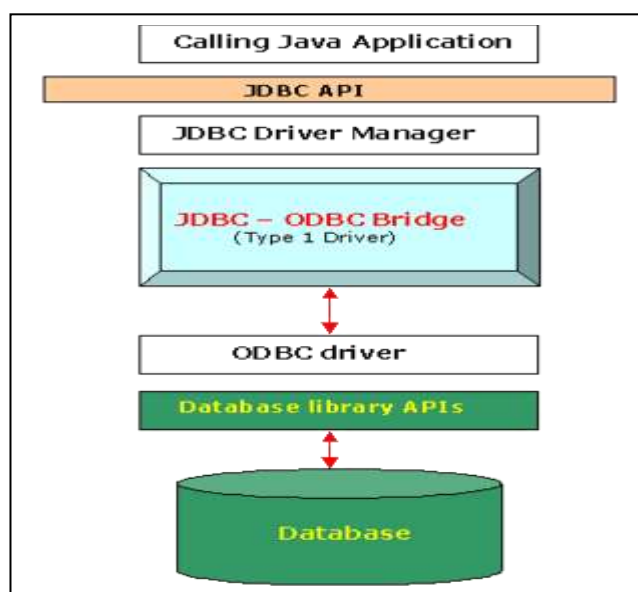
The JDBC technology drivers use one of four categories. Applications and applets can access databases through Type 1 Driver - JDBC-ODBC bridge, Type 2 Driver - Native-API Driver specification, Type 3 Driver - Network-Protocol Driver or Type 4 Driver - Native-Protocol Driver.

##### **4.2.2.1 Type 1 Driver - JDBC-ODBC bridge**

The JDBC type 1 driver is also called as the JDBC-ODBC bridge. Is the driver that depends on using the ODBC driver to connect to the database. The driver converts JDBC method calls into ODBC function calls. However, this usually uses when there is no pure-Java driver available for a particular database. As shown in the figure (4.1) below, the driver is platform-dependent to use of ODBC that depends on native libraries of the underlying operating system where the Java Virtual Machine (JVM) is running upon. The use of this driver needs other installation technologies like ODBC must be installed on the computer having the driver and the database

must support an ODBC driver. The use of this driver is not recommended if the alternatives are available such as a pure-Java driver.

Barry Cornelius (1998) stated that “Note that some ODBC binary code and in many cases database client code must be loaded on each client machine that uses this driver, so this kind of driver is most appropriate on a corporate network, or for application server code written in Java in a 3-tier architecture.”



**Figure (4.1):** Schematic of the JDBC-ODBC Bridge

#### A. Features and capabilities

- Makes translation of the query that obtained by JDBC to the corresponding ODBC query. This can be handled by ODBC driver.
- Sun provides a JDBC-ODBC Bridge driver.
- Client -> JDBC Driver -> ODBC Driver -> Database
- There is some overhead associated with the translation work to go from JDBC to ODBC.

## **I. Advantages**

- Can be accessed all databases, for which ODBC driver is installed
- A type 1 driver is easy to install

## **II. Disadvantages**

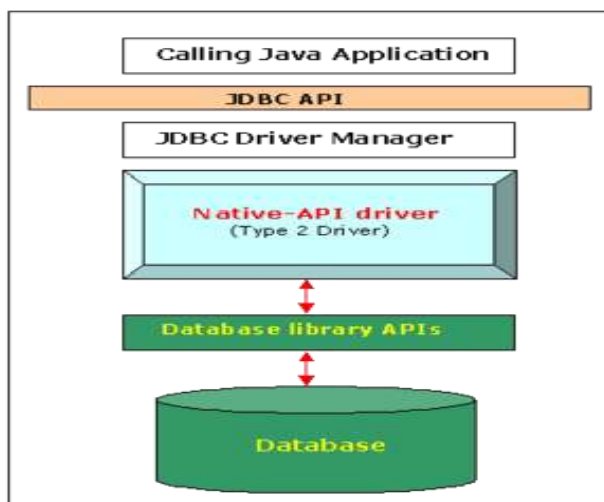
- Performance overhead since the calls have to go through the JDBC overhead bridge to the ODBC driver, then to the native db connectivity interface.
- The ODBC driver needs to be installed on the client machine.
- Considering the client-side software needed, this might not be suitable for applets.
- Will not be suitable for internet applications.
- Although type -1 is simpler, the platform specified only to Microsoft platform.

### **4.2.2.2 Type 2 Driver - Native-API Driver Specification**

The JDBC type 2 driver, is known as the Native-API driver. That is using the client-side libraries of the database. However, the driver changes JDBC method calls into native calls of the database API.

As shown in figure (4.2) below, the type 2 driver is not written entirely in Java as it interfaces with non-Java code that makes the final database calls. The driver is compiled for use with the particular operating system.

Works done by Barry Cornelius (1998) revealed that "A native-API partly-Java driver converts JDBC calls into calls on the client API for Oracle, Sybase, Informix, DB2, or other DBMS. Note that, like the bridge driver, this style of driver requires that some binary code be loaded on each client machine." Thus, the type 2 driver provides more functionality and better performance than the type 1 driver as it does not have the overhead of the additional ODBC function calls.



**Figure (4.2):** Schematic of the Native API driver

#### A. Advantages

- Better performance than Type 1, however, no needs to translate JDBC to ODBC.

#### B. Disadvantages

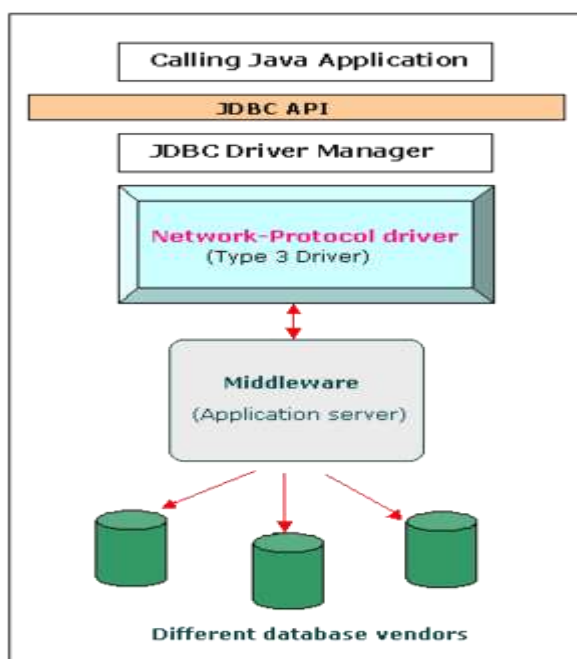
- The vendor client library needs to be installed on the client machine.
- Cannot be used in web-based application due the client side software needed.
- Not all databases have a client side library

#### 4.2.2.3 Type 3 Driver - Network-Protocol Driver

The JDBC type 3 driver is defined as the Pure Java Driver for Database Middleware as shown in figure (4.3). Is a database driver implementation that provides a process of using a middle-tier between the calling program and the database. The mission of the middle-tier (application server) is to convert JDBC calls directly or indirectly into the vendor-specific database protocol. However, is like type 4 drivers, the type 3 driver is written entirely in Java. Also is platform-

independent as the platform-related differences are taken care by the middleware. The using of middleware has additional advantage of security and firewall access.

Moreover, Barry Cornelius (1998) thinks that “the net server middleware is able to connect all its Java clients to many different databases. It depends on the number of databases the middleware has been configured to support. The specific protocol used depends on the vendor. In general, this is the most flexible JDBC alternative. It is likely that all vendors of this solution will provide products suitable for Intranet use. In order for these products to also support Internet access they must handle the additional requirements for security, access through firewalls, etc., that the Web imposes. Several vendors are adding JDBC drivers to their existing database middleware products.”



**Figure (4.3):** Schematic of the Network Protocol driver

#### A. Features and capabilities

- Follows a three tier communication approach.
- Can interface to multiple databases - Not vendor specific.



- The JDBC Client driver written in java communicates with a middleware-net-server using a database independent protocol, and then this net server translates this request into database commands for that database.
- Thus the client driver to middleware communication is database independent.
- Client -> JDBC Driver -> Middleware-Net Server -> Any Database

## **B. Advantages**

- Since the communication between client and the middleware server is database independent, there is no need for the vendor db library on the client machine. Also the client to middleware need not be changed for a new database.
- The Middleware Server (which can be a full fledged J2EE Application server) can provide typical middleware services like caching (connections, query results, and so on), load balancing, logging, auditing etc.
- Eg. for the above include JDBC driver features in Web logic.
- Can be used in internet since there is no client side software needed.
- At client side a single driver can handle any database. (It works provided the middleware supports that database!)

## **C. Disadvantages**

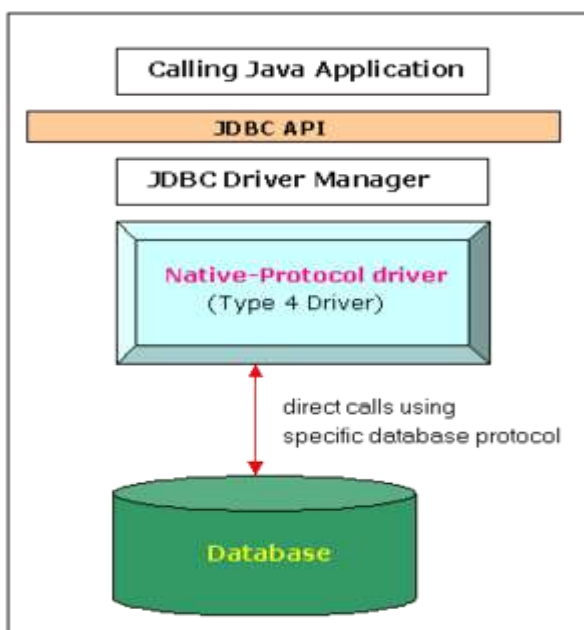
- Requires database-specific coding to be done in the middle tier.
- An extra layer added may result in a time-bottleneck. But typically this is overcome by providing efficient middleware services described above.

#### 4.2.2.4 Type 4 Driver - Native-Protocol Driver

The JDBC type 4 driver is recognized as the Direct to Database Pure Java Driver figure (4.4). Is a database driver execution that converts JDBC calls directly into the vendor-specific database protocol.

However, type 4 driver is written totally in Java and is platform independent. It is installed inside the Java Virtual Machine of the client. It provides better performance than the type 1 and 2 drivers as it does not have the overhead of conversion of calls into ODBC or database API calls. Unlike the type 3 drivers, it does not need associated software to work.

Works done by Barry Cornelius (1998) have shown that “A native-protocol all-Java driver converts JDBC calls into the network protocol used by DBMSs directly. This allows a direct call from the client machine to the DBMS server and is a practical solution for Intranet access. Since many of these protocols are proprietary the database vendors themselves will be the primary source for this style of driver. Several database vendors have these in progress.”



**Figure (4.4):** Schematic of the Native-Protocol driver

### **A. Features and capabilities**

- Type 4 drivers are entirely written in Java that communicates directly with a vendor's database, usually through socket connections. No translation or middleware layers are required, improving performance.
- The driver converts JDBC calls into the vendor-specific database protocol so that client applications can communicate directly with the database server.
- Completely implemented in Java to achieve platform independence e.g. include the widely used Oracle thin driver - `oracle.jdbc.driver.OracleDriver` which connect to `jdbc:oracle:thin` URL format.
- Client Machine -> Native protocol JDBC Driver -> Database server

### **B. Advantages**

- These drivers don't translate the requests into an intermediary format (such as ODBC), nor do they need a middleware layer to service requests. Thus the performance may be considerably improved.
- All aspects of the application to database connection can be managed within the JVM; this can facilitate easier debugging.

### **C. Disadvantage**

- At client side, a separate driver is needed for each database

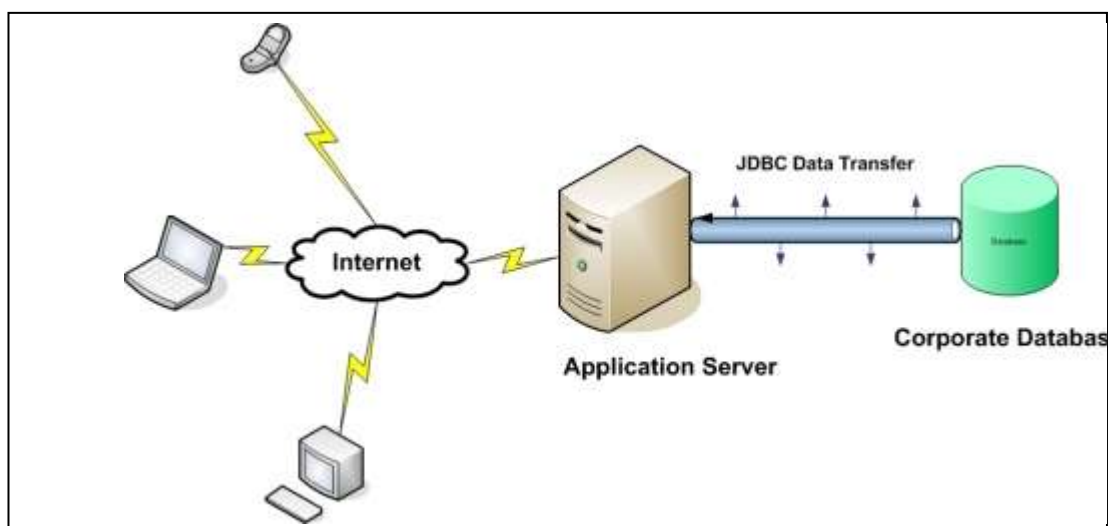
### **D. JDBC Security feature (encryption/decryption)**

Generally the JDBC standard is used to make connection between Java applications and database. We mostly have web applications running in a web container. These web applications make JDBC connections to the databases to retrieve data for the applications. However, we have to think how secure these JDBC calls are. Like any other insecure network protocol call.

Before we go any further let us see some of the security threats that face normal JDBC connections

- Eavesdropping and Data Theft
- Data Tampering
- Falsifying User Identities
- Password-Related Threats

Data flowing over the network is prone to network sniffers, be it inside or outside your company. Anyone can pick up these data packets and tamper with them. The figure (4.5) below shows the data flows over Internet and network.



**Figure (4.5):** Data flows over the network

To solve these security challenges some vendors like Oracle Corporation released JDBC that support encryption. In other words, it means that the JDBC calls will be transparently encrypted using standard encryption algorithms, thereby making the data transfer secure. Secondly the data will be hashed to form message digests, thereby preventing data tampering, Franklin (2008).

A published from JDBC Developer's Guide and Reference (2002) revealed that if we are using one of the Oracle JDBC OCI drivers, which presume a thick-

client setting with an Oracle client installation, we can enable or disable data encryption or integrity and set related parameters as we would in any Oracle client situation, through settings in the SQLNET.ORA file on the client machine.

Also Microsoft Corporation (2007) made knowledge that “The SQL Server 2005 JDBC Driver is available to all SQL Server users at no additional charge, and provides access to SQL Server 2000 and SQL Server 2005 from any Java application, application server, or Java-enabled applet. This driver is a Type 4 JDBC driver that provides database connectivity through the standard JDBC application program interfaces (APIs) available in J2EE (Java2 Enterprise Edition).” However, this latest version of the JDBC driver provides an adaptive buffering feature to retrieve any kind of large-value data without the overhead of server cursors. In addition, this release includes support for Secure Sockets Layer (SSL) encryption and tightly coupled distributed transactions.

#### 4.2.3 Comparison of the 4 Types of JDBC Drivers

**Table (4.1):** JDBC Drivers types contrast

<b>JDBC Drivers</b>	<b>Pros</b>	<b>Cons</b>
Type 1: JDBC- ODBC bridge drivers	integrated into JDK v 1.1	<ul style="list-style-type: none"> <li>• Requires ODBC manager on the client;</li> <li>• The ODBC driver on each of the client's needs to be configured.</li> </ul>
Type 2: Native API Partially- JAVA	allows the programmer to fully utilize the speed and power that comes from the use of the API specifically developed for the	<ul style="list-style-type: none"> <li>• Driver is DBMS-dependent;</li> <li>• Requires a vendor-supplied DLL</li> </ul>

drivers	DBMS	(Dynamic Link Library) to be installed in the client's JAVA library path.
Type 3: Net-Protocol All-JAVA drivers	DBMS-independent; allows the most flexible multi-server configuration	<ul style="list-style-type: none"> <li>• Requires a vendor-supplied intermediate server;</li> <li>• Configuration of the intermediate server</li> </ul>
Type 4: Native Protocol All-JAVA drivers	<ul style="list-style-type: none"> <li>• Allows a direct call from the client to the database, without the need of client pre configuration</li> <li>• Allow encryption with the latest versions of JDBC</li> </ul>	<ul style="list-style-type: none"> <li>• Requires the use of protocols proprietary to the DBMS,;</li> <li>• Need to load a different driver for each DBMS that it needs to access.</li> </ul>

#### 4.2.4 General Advantages of JDBC Technology

1. Leverage Existing Enterprise Data: With JDBC technology, businesses are not locked in any proprietary architecture, and can continue to use their installed databases and access information easily -- even if it is stored on different database management systems.
2. Simplified Enterprise Development: The combination of the Java API and the JDBC API makes application development easy and economical. JDBC hides the complexity of many data access tasks, doing most of the "heavy lifting "for the programmer behind the scenes. The JDBC API is simple to learn, easy to deploy, and inexpensive to maintain.
3. Zero Configuration for Network Computers: With the JDBC API, no configuration is required on the client side. With a driver written in the

Java programming language, all the information needed to make a connection is completely defined by the JDBC URL or by a Data Source object registered with a Java Naming and Directory Interface (JNDI) naming service. Zero configuration for clients supports the network computing paradigm and centralizes software maintenance.

4. **Key Features Full Access to Metadata:** The JDBC API provides metadata access that enables the development of sophisticated applications that need to understand the underlying facilities and capabilities of a specific database connection.
5. **Database Connection Identified by URL:** JDBC technology exploits the advantages of Internet-standard URLs to identify database connections. The JDBC API includes an even better way to identify and connect to a data source, using a Data Source object that makes code even more portable and easier to maintain.

### **4.3 Microsoft Open Database Connectivity (ODBC)**

Microsoft developed the ODBC interface as a means of providing applications with a single application programming interface (API) through which to access data stored in a wide variety of DBMSs. ODBC is designed to give applications the ability to access different database management systems with the same source code. The data source is not necessarily a DBMS; an application can even access text-files or Excel documents using ODBC. Today ODBC is a very widespread API with hundreds of ODBC-enabled applications.

However, the ODBC interface allows highest interoperability; an application can use data in various DBMSs through a single interface. Moreover, that application will be independent of any DBMS from which it accesses data. Users of the application can add software components called drivers, which interface between an application and a specific DBMS (Microsoft Corporation, 2009).

On the other hand, ODBC is a specification for a database application programming interface API. This API is independent of any one DBMS or operating system. The ODBC API is based on the CLI specifications from Open Group and ISO/IEC. ODBC 3.x fully implements both of these specifications — earlier versions of ODBC were based on preliminary versions of these specifications but did not fully implement them — and adds features commonly needed by developers of screen-based database applications, such as scrollable cursors.

The functions in the ODBC API are implemented by developers of DBMS-specific drivers. Applications call the functions in these drivers to access data in a DBMS-independent manner. A Driver Manager manages communication between applications and drivers.

“Although Microsoft provides a driver manager for computers running Microsoft Windows 95 and later, has written several ODBC drivers, and calls ODBC functions from some of its applications, anyone can write ODBC applications and drivers. In fact, the vast majority of ODBC applications and drivers available today are written by companies other than Microsoft. Furthermore, ODBC drivers and applications exist on the Macintosh and a variety of UNIX platforms (Microsoft Corporation, 2009a).”

In order to assist application and driver developers, Microsoft produces an ODBC Software Development Kit (SDK) for computers running Windows 95. Which that provides the driver manager, installer DLL, test tools, and sample applications. However, Microsoft has support Software to port these SDKs to the Macintosh and a variety of UNIX platforms as mentioned by (Microsoft Corporation 2009a).

The main objective of the ODBC is designed to representation database capabilities, not supplement them. “Thus, application writers should not expect that using ODBC will suddenly transform a simple database into a fully featured relational database engine. Nor are driver writers expected to implement



functionality not found in the underlying database. An exception to this is that developers who write drivers that directly access file data (such as data in an Xbase file) are required to write a database engine that supports at least minimal SQL functionality. Another exception is that the ODBC component of the Windows SDK, formerly included in the Microsoft Data Access Components (MDAC) SDK, provides a cursor library that simulates scrollable cursors for drivers that implement a certain level of functionality (Microsoft Corporation, 2009a).”

#### **4.3.1 How ODBC Works?**

To use the ODBC, three components are needed: ODBC client, ODBC driver, and a DBMS server (ex. Microsoft Access, SQL Server, Oracle, and FoxPro). Firstly, the ODBC client will use a command (referred to as "ODBC") to interact (requesting and/or sending data) with the DBMS server (back-end). However, the DBMS server will not understand the command by the ODBC client yet, as the command has yet to be processed through the ODBC driver (front-end). So then, the ODBC driver will decode the command that can be processed by the ODBC server and be sent there. The ODBC server will then respond back to the ODBC driver which will translate the final output to the ODBC client (Tech-FAQ, 2008).”

Moreover, the ODBC directs this by inserting a mediator, called a database driver, between an application and the DBMS. The purpose of this layer is to translate the application's data queries into commands that the DBMS can use them. For this to work, both the application and the DBMS must be ODBC-compliant -- that is, the application must be capable of issuing ODBC commands and the DBMS must be capable of responding to them.

The ODBC Specification offers a procedural application programming interface API for using SQL queries to access data. An implementation of ODBC will contain one or more applications, a core ODBC "Driver Manager" library, and one or more "database drivers.” The Driver Manager, independent of the applications

and DBMS, acts as an "interpreter" between the applications and the database drivers, whereas the database drivers contain the DBMS-specific details. Thus a programmer can write applications that use standard types and features without concern for the specifics of each DBMS that the applications may encounter. Likewise, database driver implementers only need to know how to attach to the core library. This makes ODBC modular.

The ODBC provides the standard of ubiquitous data access because hundreds of ODBC drivers exist for a large variety of data sources. ODBC runs with a variety of operating systems and drivers exist for non-relational data such as MS-Access, dBase, DB2, Excel, and Text. As well as it operates with relational database, for instance Oracle. Through these Call Level Interface (CLI) specifications of the SQL Access Group, the ODBC allows a neutral way of accessing the data stored in personal computers and various databases. Because ODBC dates back to 1992, it offers connectivity to a wider variety of data sources than other data-access APIs.

The strength of ODBC is that by providing a common data access interface, it allows independent software providers and parties to not have to learn multiple application programming interfaces. To simply put, with ODBC, applications can simultaneously access, view, and modify database from many and quite diverse databases. This is because the ODBC "re-codes" the SQL queries so that it would be readable by the various different databases.

#### **4.3.2 ODBC Drivers**

“Drivers are libraries that implement the functions in the ODBC API. Each is specific to a particular DBMS; for example, a driver for Oracle cannot directly access data in an Informix DBMS. Drivers expose the capabilities of the underlying DBMSs; they are not required to implement capabilities not supported by the DBMS. For example, if the underlying DBMS does not support outer joins, then neither should the driver. The only major exception to this is that drivers for DBMSs that do

not have stand-alone database engines, such as Xbase, must implement a database engine that at least supports a minimal amount of SQL (Microsoft Corporation, 2009a).”

#### 4.3.2.1 Driver Tasks

The following specific tasks can be performed by drivers:

- Connecting to and disconnecting from the data source.
- Checking for function errors not checked by the Driver Manager.
- Initiating transactions; this is transparent to the application.
- Submitting SQL statements to the data source for execution. The driver must modify ODBC SQL to DBMS-specific SQL; this is often limited to replacing escape clauses defined by ODBC with DBMS-specific SQL.
- Sending data to and retrieving data from the data source, including converting data types as specified by the application.
- Mapping DBMS-specific errors to ODBC SQLSTATEs.

#### 4.3.2.2 Driver Architecture

Driver architecture is classified into two categories, however, depending on which software processes SQL statements (Microsoft Corporation, 2009).

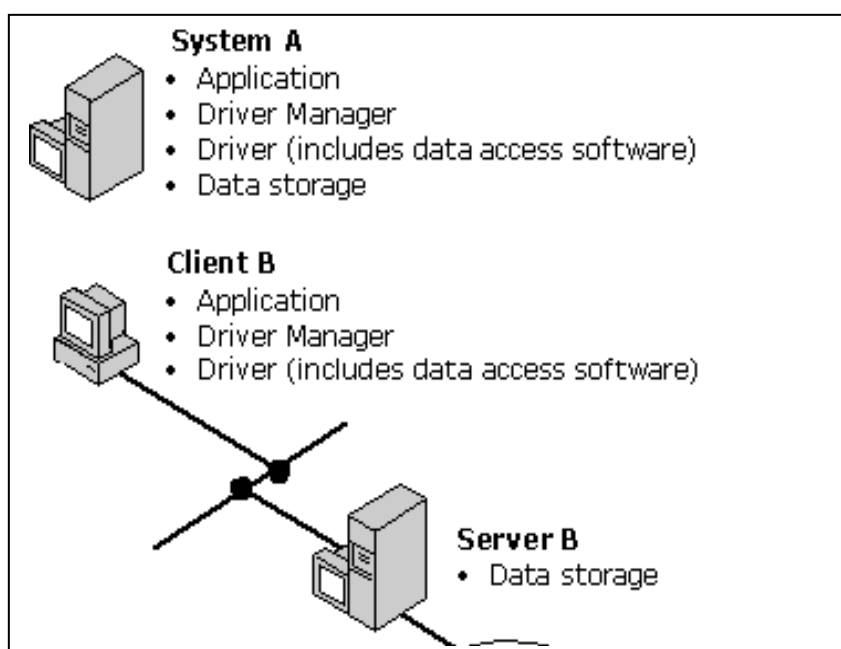
- **File-Based Drivers:** The driver accesses the physical data directly. In this case, the driver acts as both driver and data source; that is, it processes ODBC calls and SQL statements. For example, dBASE drivers are file-based drivers because dBASE does not provide a stand-alone database engine the driver can use.
- **DBMS-Based Drivers:** The driver accesses the physical data through a separate database engine. In this case the driver processes only ODBC calls; it passes SQL statements to the database engine for processing. For example, Oracle drivers are DBMS-based drivers because Oracle has a

stand-alone database engine the driver uses. Where the database engine resides is immaterial. It can reside on the same machine as the driver or a different machine on the network; it might even be accessed through a gateway.

#### A. File-Based Drivers

File-based drivers are used with data sources such as dBASE that do not provide a stand-alone database engine for the driver to use. These drivers access the physical data directly and must implement a database engine to process SQL statements. As a standard practice, the database engines in file-based drivers implement the subset of ODBC SQL defined by the minimum SQL conformance level (Microsoft Corporation, 2009).

The following diagrams show two different configurations of file-based drivers, one in which the data resides locally and the other in which it resides on a network file server.



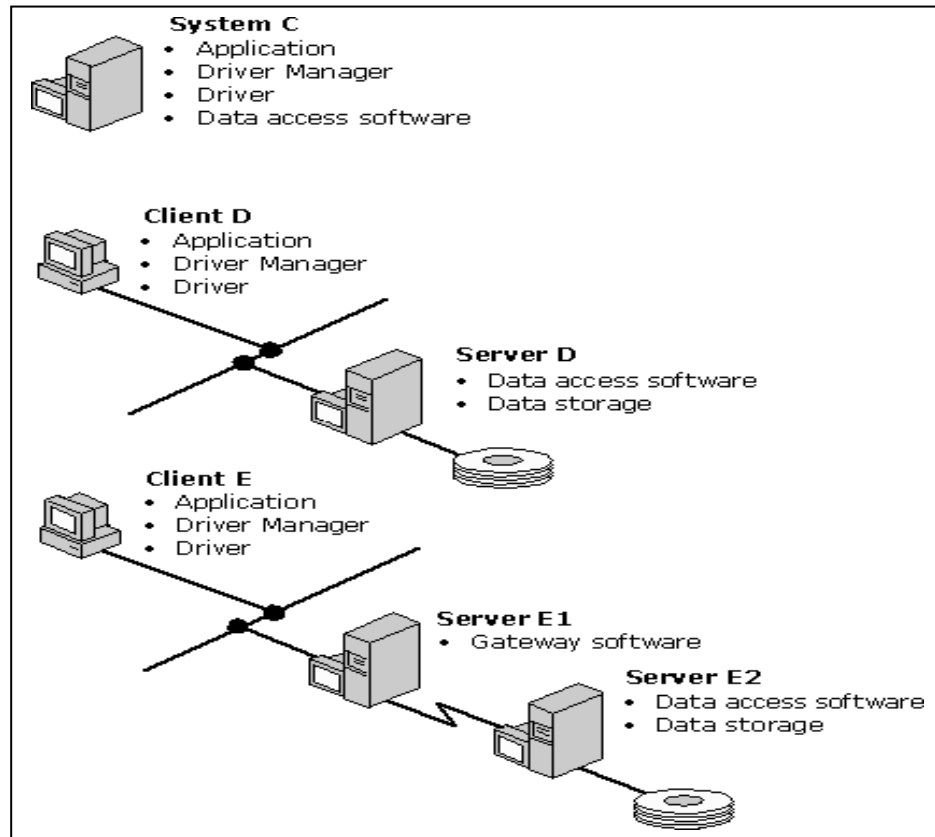
**Figure (4.6):** Different configurations of file-based drivers

## **B. DBMS-Based Drivers**

DBMS-based drivers are used with data sources such as Oracle or SQL Server that provide a stand-alone database engine for the driver to use. These drivers access the physical data through the stand-alone engine; that is, they submit SQL statements to and retrieve results from the engine (Microsoft Corporation, 2009a).

Because DBMS-based drivers use an existing database engine, they are usually easier to write than file-based drivers. Although a DBMS-based driver can be easily implemented by translating ODBC calls to native API calls, this results in a slower driver. A better way to implement a DBMS-based driver is to use the underlying data stream protocol, which is usually what the native API does. For example, a SQL Server driver should use TDS (the data stream protocol for SQL Server) rather than DB Library (the native API for SQL Server). An exception to this rule is when ODBC is the native API. For example, Watcom SQL is a stand-alone engine that resides on the same machine as the application and is loaded directly as the driver (Microsoft Corporation, 2009).

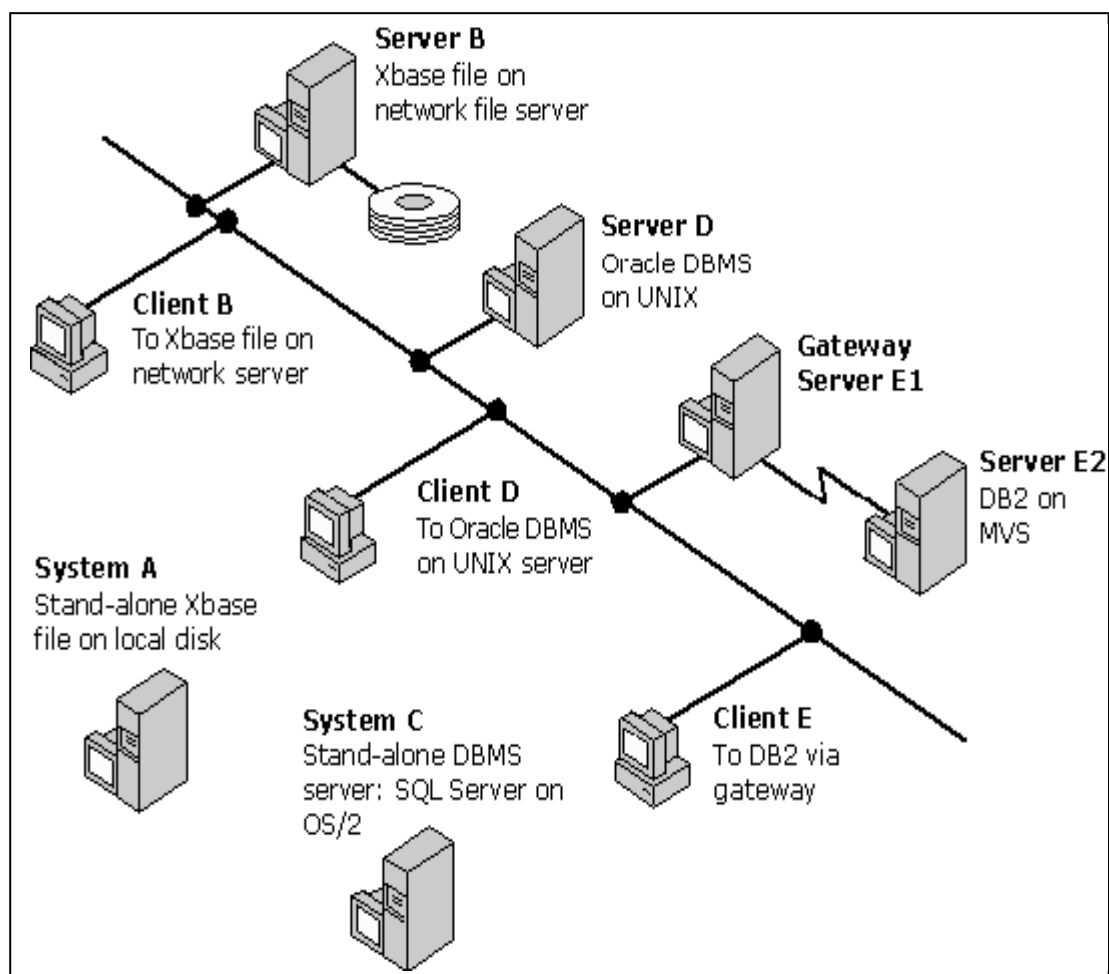
DBMS-based drivers act as the client in a client/server configuration where the data source acts as the server. In most cases, the client (driver) and server (data source) reside on different machines, although both could reside on the same machine running a multitasking operating system. A third possibility is a gateway, which sits between the driver and data source. A gateway is a piece of software that causes one DBMS to look like another. For example, applications written to use SQL Server can also access DB2 data through the Micro Decision ware DB2 Gateway; this product causes DB2 to look like SQL Server (Microsoft Corporation, 2009). The following illustration shows three different configurations of DBMS-based drivers.



**Figure (4.7):** Different configurations of DBMS-based drivers.

### C. Network Example

This illustration shows how each of the preceding configurations could appear in a single network (Microsoft Corporation, 2009a).



**Figure (4.8):** Single network configuration

### 4.3.2.3 Other Driver Architectures

Some ODBC drivers do not strictly conform to the architecture described previously. This might be because the drivers perform duties other than those of a traditional ODBC driver, or are not drivers in the normal sense.

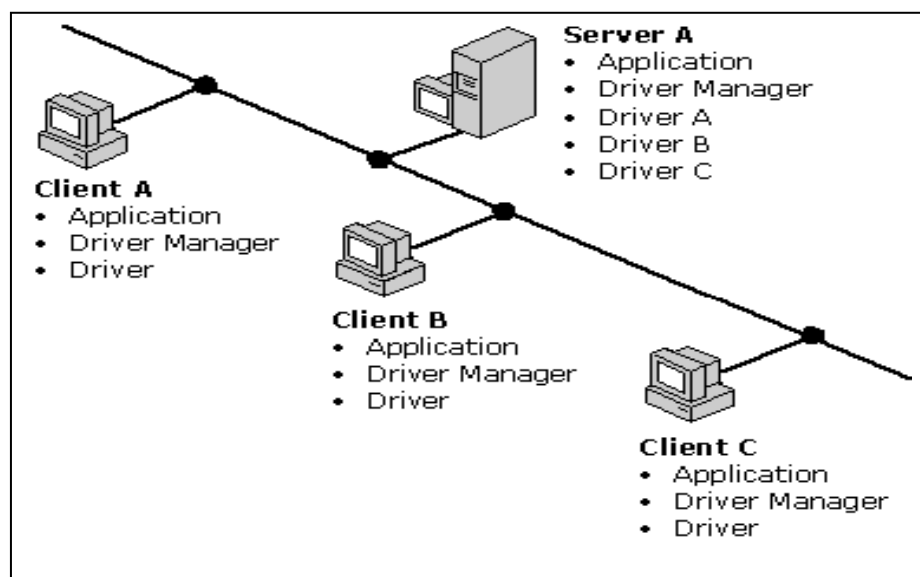
#### A. Driver as a Middle Component

The ODBC driver may reside between the Driver Manager and one or more other ODBC drivers. When the driver in the middle is capable of working with multiple data sources, it acts as a dispatcher of ODBC calls (or appropriately

translated calls) to other modules that actually access the data sources. In this architecture, the driver in the middle is taking on some of the role of a Driver Manager.

## B. ODBC on the Server

ODBC drivers can be installed on a server so that they can be used by applications on any of a series of client machines. In this architecture (see the following illustration Figure 3.9 , a Driver Manager and a single ODBC driver are installed on each client, and another Driver Manager and a series of ODBC drivers are installed on the server. This allows each client access to a variety of drivers used and maintained on the server.



**Figure (4.9):** Installation of ODBC on the Server

One advantage of this architecture is efficient software maintenance and configuration. Drivers need only be updated in one place: on the server. By using system data sources, data sources can be defined on the server for use by all clients. The data sources need not be defined on the client. Connection pooling can be used to streamline the process by which clients connect to data sources.



The driver on the client is usually a very small driver that transfers the Driver Manager call to the server. Its footprint can be significantly smaller than the fully functional ODBC drivers on the server. In this architecture, client resources can be freed if the server has more computing power. In addition, the efficiency and security of the entire system can be enhanced by installing backup servers and performing load balancing to optimize server use.

#### **4.3.2.4 Microsoft ODBC Desktop Database Drivers**

ODBC is an API that uses Structured Query Language (SQL) as the database access language. You can access a wide variety of database management systems (DBMSs) with the same ODBC source code that is directly incorporated into an application's source code. With the Microsoft ODBC Desktop Database Drivers, a user of an ODBC-enabled application can open, query, and update a desktop database through the ODBC interface.

The Microsoft ODBC Desktop Database Drivers provide access to the following types of data sources:

- Microsoft Access
- Microsoft Excel
- Paradox
- dBASE
- Text
- Desktop Database Drivers Architecture

These drivers are designed for use on Microsoft Windows 95 or later, or Windows NT 4.0 and Windows 2000. Only 32-bit applications are supported on Windows 95 or later; 16-bit and 32-bit applications are supported on Windows NT 4.0 and Windows 2000.

#### **4.3.2.5 ODBC Driver for Oracle**

The Microsoft ODBC Driver for Oracle allows you to connect your ODBC-compliant application to an Oracle database. The ODBC Driver for Oracle conforms to the Open Database Connectivity (ODBC) specification described in the ODBC Programmer's Reference. It allows access to PL/SQL packages, XA/DTC integration, and Oracle access from within Internet Information Services (IIS).

Oracle RDBMS is a multiuser relational database management system that runs with various workstation and minicomputer operating systems. IBM-compatible computers running Microsoft Windows can communicate with Oracle database servers over a network. Supported networks include Microsoft LAN Manager, NetWare, VINES, DECnet, and any network that supports TCP/IP.

The ODBC Driver for Oracle enables an application to access data in an Oracle database through the ODBC interface. The driver can access local Oracle databases or it can communicate with the network through SQL\*Net. To access Oracle data, the following components are required:

- The ODBC Driver for Oracle
- An Oracle RDBMS database
- Oracle Client Software

Additionally, for remote connections:

- A network that connects the computers that run the driver and the database. The network must support SQL\*Net connections.

#### **4.3.2.6 Oracle Form Developer and ODBC**

Moreover, The Oracle Open Client Adapter for ODBC (OCA) allows Forms Developer and Reports Developer on Microsoft Windows 95 and Windows NT to

access ODBC-compliant data sources through ODBC drivers. Using the Oracle Open Client Adapter, an application can access different data sources in a consistent manner. This allows an application developer to build an application that can run unmodified against one of several databases. Alternatively, the application can be targeted at a specific database, and take advantage of features particular to that system.

#### **4.3.2.7 Visual FoxPro ODBC Driver**

Microsoft Visual FoxPro is a powerful object-oriented environment for database construction and application development. The Microsoft Visual FoxPro ODBC Driver enables applications to open, query, and update data in Visual FoxPro and earlier versions of FoxPro through the Open Database Connectivity (ODBC) interface.

#### **4.3.2.8 ODBC Architecture**

The ODBC architecture has four components:

- Application performs processing and calls ODBC functions to submit SQL statements and retrieve results.
- Driver Manager Loads and unloads drivers on behalf of an application. Processes ODBC function calls or passes them to a driver.
- Driver Processes ODBC function calls, submits SQL requests to a specific data source, and returns results to the application. If necessary, the driver modifies an application's request so that the request conforms to syntax supported by the associated DBMS.
- Data Source Consists of the data the user wants to access and its associated operating system, DBMS, and network platform (if any) used to access the DBMS.

### 4.3.3 Encryption

Encryption is a method that is used for encoding the data so that other programs cannot read it. Some DBMSs contain an internal encryption mechanism, while other relies on the operating system or third-party programs. Such as the Mac OS X version 10.1.x ("Puma") and 10.2.x ("Jaguar") facilitate to Use strong encryption of data - Enable SSL encryption of data between driver and database as revealed by OpenLink Software (2003).

Moreover, the SQL Server ODBC driver now lets you protect the confidentiality of SQL Server data with the following encryption algorithms (Easysoft Limited, 2009).

- AES, 3DES, DES Encryption Keep data transmitted between Linux/Unix and SQL Server secure with industry standard encryption.
- Transport Layer Security The latest and most secure version of SSL.
- SSL Cipher Suites Enforce security policy with SSL cipher suites.

### 4.3.4 ODBC Advantages:-

ODBC implements the following features to call-level interface specifications that exist in the ISO/IEC and Open Group CLI standards (Microsoft Corporation, 2009).

- Multi row fetches by a single function call
- Binding to an array of parameters
- Bookmark support including fetching by bookmark, variable-length bookmarks, and bulk update and delete by bookmark operations on discontinuous rows
- Row-wise binding
- Binding offsets

- Support for batches of SQL statements, either in a stored procedure or as a sequence of SQL statements executed through `SQLExecute` or `SQLExecDirect`
- Exact or approximate cursor row counts
- Positioned update and delete operations and batched updates and deletes by function call (`SQLSetPos`)
- Catalog functions that extract information from the information schema without the need for supporting information schema views
- Escape sequences for outer joins, scalar functions, date time literals, interval literals, and stored procedures
- Code-page translation libraries
- Reporting of a driver's ANSI-conformance level and SQL support
- On-demand automatic population of implementation parameter descriptor
- Enhanced diagnostics and row and parameter status arrays
- Date time, interval, numeric/decimal, and 64-bit integer application buffer types
- Asynchronous execution
- Stored procedure support, including escape sequences, output parameter binding mechanisms, and catalogue functions
- Connection enhancements including support for connection attributes and attribute browsing

#### **4.3.5 ODBC Disadvantages**

Works had done by Brooke Barnabe (2007) revealed that “although ODBC made application-database interaction easier, it still has some disadvantages. One of the most common, but incorrect, assumptions is that ODBC is slower due to all of its components. Fortunately, it has been refined enough to work just as well as native APIs. ODBC still employs some limitations. It can only be used for relational databases and as industry has grown towards object-oriented programming and design, ODBC is being increasingly neglected for its lack of OO support (even

though it supports JAVA objects). Also, it works in its optimal state when data types can be represented in rectangular, two-dimensional or tabular format which is not conducive to objects. Another complaint about ODBC is that vendor provided APIs are much faster and more direct than ODBC. ODBC sidetracks access that might otherwise have the capability to be direct. Because ODBC is created by Microsoft, they are the only group with control over the specification of the API. They determine when it gets updated and what features are available for use. Some people complain that Microsoft uses ODBC for its own personal gain.”

#### **4.4 ADO**

ActiveX Data Objects (ADO) is a high-level, easy-to-use interface to OLE DB. OLE DB is a low-level, high-performance interface to a variety of data stores. Both ADO and OLE DB can work with relational (tabular) and nonrelational (hierarchical or stream) data (Microsoft corporation, 2009b).

ADO provides a layer of abstraction between your client or middle-tier application and the low-level OLE DB interfaces. ADO uses a small set of Automation objects to provide a simple and efficient interface to OLE DB. This interface makes ADO a good choice for developers in higher level languages, such as Visual Basic and VBScript, who want to access data without having to learn the intricacies of COM and OLE DB.

ADO sits on top of and uses OLE DB, which provides the underlying information access. Any data provider that offers an OLE DB interface, for example SQL Server 2000, Microsoft Exchange or Active Directory, can be accessed through ADO. As ADO uses the Common Object Model (COM) automation interface, it is available from all leading Rapid Application Development (RAD) tools, database tools, and languages. ADO is a collection of COM objects that provide a common interface, optimized data access, cross language support and support numerous development scenarios.

Kenneth Lassen (1999) from Microsoft said that “ActiveX Data Objects (ADO) provide access to a rich variety of data sources through an OLE DB Provider. Typically, the OLE DB Provider is the OLE DB Provider for ODBC Drivers, effectively granting access to any data source having an ODBC Driver. ActiveX Data Objects are language-neutral object models that expose data raised by an underlying OLE DB Provider. The most commonly used OLE DB Provider is the OLE DB.

It is positioned as a successor to Microsoft's earlier object layers for accessing data sources, including RDO (Remote Data Objects) and DAO (Data Access Objects). ADO was introduced by Microsoft in October 1996 (Wikipedia, 2008).

#### **4.4.1 ADO Fundamentals**

ADO gives developers a powerful, logical object model for programmatically accessing, editing, and updating data from a wide variety of data sources through OLE DB system interfaces. The most common usage of ADO is to query a table or tables in a relational database, retrieve and display the results in an application, and perhaps let users make and save changes to the data. Other tasks include the following (Microsoft corporation, 2009b):

- Querying a database using SQL and displaying the results.
- Accessing information in a file store over the Internet.
- Manipulating messages and folders in an e-mail system.
- Saving data from a database into an XML file.
- Executing commands described with XML and retrieving an XML stream.
- Saving data into a binary or XML stream.
- Allowing a user to review and change data in database tables.
- Creating and reusing parameterized database commands.
- Executing stored procedures.

- Dynamically creating a flexible structure, which is named a Recordset, to hold, navigate, and manipulate data.
- Performing transactional database operations.
- Filtering and sorting local copies of database information based on run-time criteria.
- Creating and manipulating hierarchical results from databases.
- Binding database fields to data-aware components.
- Creating remote, disconnected Recordsets.

ADO exposes a wide variety of options and settings to provide such flexibility. Therefore, it is important to take a methodical approach to learning how to use ADO in an application, breaking down each of your goals into manageable pieces.

However, languages such as Delphi and C++ Builder, development environments from Microsoft rival Borland Software Corporation, also allow the use of ADO to access various databases. In the newer programming framework of .NET, Microsoft also presented an upgraded version of ADO called ADO.NET. Its object structure is quite different from that of traditional ADO.

Some basic steps are required in order to be able to access and manipulate data using ADO:

1. Create a connection object to connect to the database.
2. Create a recordset object in order to receive data in.
3. Open the connection
4. Populate the recordset by opening it and passing the desired table name or SQL statement as a parameter to open function.
5. Do all the desired searching/processing on the fetched data.
6. Commit the changes you made to the data (if any) by using Update or UpdateBatch methods.
7. Close the recordset
8. Close the connection



#### **4.4.2 ADO Advantages**

Barry Dorrans. (2008) revealed that the common interface to accessing data, independent of its location provides some main advantages:-

- Portability between applications and languages
- Common Code Base
- Reliability
- Easier debugging and documentation
- Ease of use
- High speed
- Low memory requirements
- Small size and disk footprint
- Wide variety of drivers available. You can use ADO to talk to databases such as: Access, Microsoft SQL Server, Oracle, MySQL, Postgres SQL, Sybase, Firebird, Interbase and DB.

#### **4.4.3 ADO Disadvantage**

Mayukh Bose. (2004) mentioned the disadvantage as follows:-

- A program that uses ADO to connect to the database can only be on a Microsoft operating system currently. The database itself can be on a non-microsoft platform though.

#### **4.5 ADO.NET**

ADO.NET provides consistent access to data sources such as Microsoft SQL Server, as well as data sources exposed through OLE DB and XML. Data-sharing consumer applications can use ADO.NET to connect to these data sources and retrieve, manipulate, and update data.

In other words, the technology to connect to and manipulate data in data stores in .NET is called ADO.NET (ActiveX Data Objects). ADO.NET uses a disconnected strategy. Instead of keeping a connection open, data is transferred to the client, the connection closed and the data is manipulated. Then a new connection is opened and the manipulated data is uploaded. Some database managing systems have the capability use XML for data exchange. ADO.NET is well prepared for this.

ADO.NET cleanly factors data access from data manipulation into discrete components that can be used separately or in tandem. ADO.NET includes .NET Framework data providers for connecting to a database, executing commands, and retrieving results. Those results are either processed directly, or placed in an ADO.NET DataSet object in order to be exposed to the user in an ad-hoc manner, combined with data from multiple sources, or remoted between tiers. The ADO.NET DataSet object can also be used independently of a .NET Framework data provider to manage data local to the application or sourced from XML (.NET Framework Developer's Guide, 2009).

#### **4.5.1 ADO.NET Architecture**

The ADO.NET components have been designed to factor data access from data manipulation. There are two central components of ADO.NET that accomplish this: the DataSet, and the .NET Framework data provider, which is a set of components including the Connection, Command, DataReader, and DataAdapter objects.

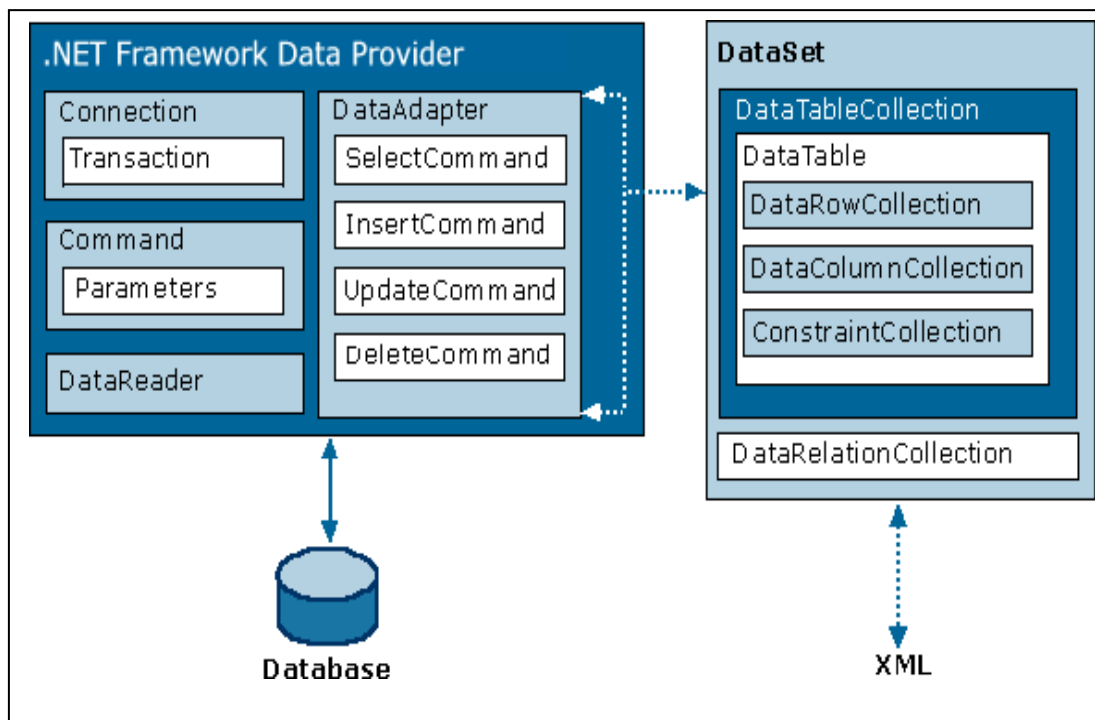
The ADO.NET DataSet is the core component of the disconnected architecture of ADO.NET. The DataSet is explicitly designed for data access independent of any data source. As a result it can be used with multiple and differing data sources, used with XML data, or used to manage data local to the application. The DataSet contains a collection of one or more DataTable objects made up of rows

and columns of data, as well as primary key, foreign key, constraint, and relation information about the data in the DataTable objects.

The other core element of the ADO.NET architecture is the .NET Framework data provider, whose components are explicitly designed for data manipulation and fast, forward-only, read-only access to data. The Connection object provides connectivity to a data source. The Command object enables access to database commands to return data, modify data, run stored procedures, and send or retrieve parameter information. The DataReader provides a high-performance stream of data from the data source. Finally, the DataAdapter provides the bridge between the DataSet object and the data source. The DataAdapter uses Command objects to execute SQL commands at the data source to both load the DataSet with data, and reconcile changes made to the data in the DataSet back to the data source.

We can write .NET Framework data providers for any data source. The .NET Framework ships with two .NET Framework data providers: the .NET Framework Data Provider for SQL Server and the .NET Framework Data Provider for OLE DB.

The following figure (3.10) illustrates the components of ADO.NET architecture (.NET Framework Developer's Guide, 2009).



**Figure (4.10):** ADO.NET architecture

#### 4.5.2 ADO.NET Encryption

J.D. Meier *et al.* (2005) from Microsoft Corporation announced that “If sensitive data must be stored, then a strong symmetric encryption algorithm such as AES is used to encrypt it. DPAPI is used to protect symmetric encryption keys.”

#### 4.5.3 ADO.NET Advantages

1. Performance – there is no doubt that ADO.NET is extremely fast.
2. Optimized SQL Provider – in addition to performing well under general circumstances, ADO.NET includes a SQL Server Data Provider that is highly optimized for interaction with SQL Server
3. XML Support (and Reliance) – everything you do in ADO.NET at some point will boil down to the use of XML
4. Disconnected Operation Model – the core ADO.NET class, the DataSet, operates in an entirely disconnected fashion. However, the disconnected

model allows for the DataSet class to be unaware of the origin of its data, an unlimited number of supported data sources can be plugged into code without any hassle in the future.

#### **4.5.4 ADO.NET Disadvantages**

1. Managed-Only Access – for a few obvious reasons, and some far more technical, you cannot utilize the ADO.NET architecture from anything but managed code. This means that there is no COM interoperability allowed for ADO.NET. Therefore, in order to take advantage of the advanced SQL Server Data Provider and any other feature like DataSets, XML internal data storage, etc, your code must be running under the CLR.
2. Only Three Managed Data Providers (so far) – unfortunately, if you need to access any data that requires a driver that cannot be used through either an OLEDB provider or the SQL Server Data Provider, then you may be out of luck. However, the good news is that the OLEDB provider for ODBC is available for download from Microsoft.

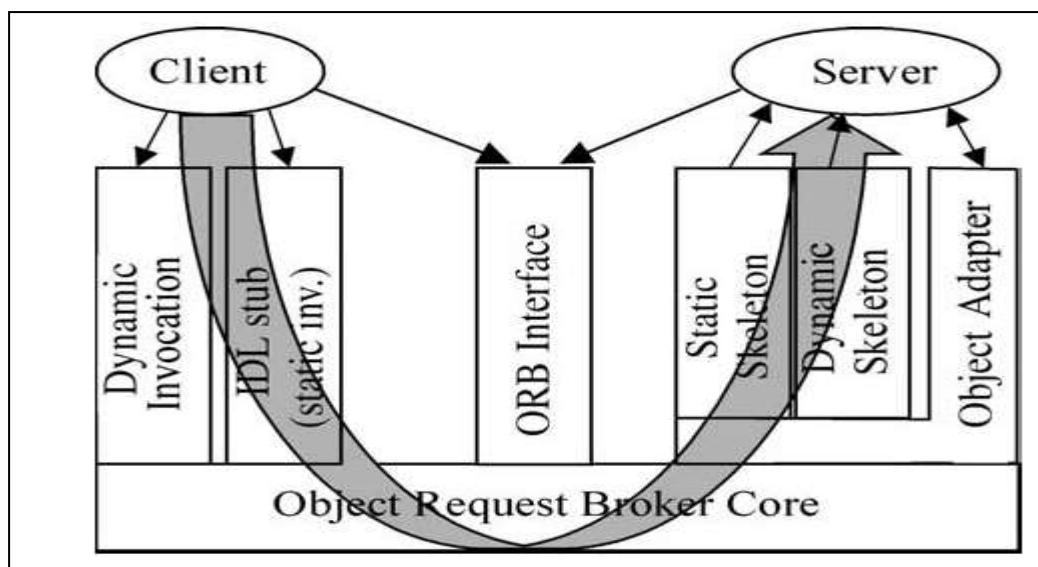
#### **4.6 CORBA**

The Common Object Request Broker Architecture (CORBA) is a standard developed by the Object Management Group (OMG) to provide interoperability among distributed objects. CORBA is the world's leading middleware solution enabling the exchange of information, independent of hardware platforms, programming languages, and operating systems. CORBA is essentially a design specification for an Object Request Broker (ORB), where an ORB provides the mechanism required for distributed objects to communicate with one another, whether locally or on remote devices, written in different languages, or at different locations on a network.

The interface language for CORBA programs is the Interface Definition Language (IDL). The IDL syntax is essentially that of C++, except that IDL defines interfaces rather than implementations. In a CORBA application, the IDL is written first, and then compiled into code in one of the supported languages. The elements defined in the IDL are then implemented in that language using the generated code as the basis. IDL has four primary elements: modules, interfaces, operations, and attributes (Arnold, B., and, 1998).

The CORBA Interface Definition Language, or IDL, allows the development of language and location-independent interfaces to distributed objects. Using CORBA, application components can communicate with one another no matter where they are located, or who has designed them. CORBA provides the location transparency to be able to execute these applications.

CORBA is often described as a "software bus" because it is a software-based communications interface through which objects are located and accessed. The illustration below identifies the primary components seen within a CORBA implementation. Data communication from client to server is accomplished through a well-defined object-oriented interface. The Object Request Broker (ORB) determines the location of the target object, sends a request to that object, and returns any response back to the caller. Through this object-oriented technology, developers can take advantage of features such as inheritance, encapsulation, polymorphism, and runtime dynamic binding. These features allow applications to be changed, modified and re-used with minimal changes to the parent interface. The illustration figure (4.11) below identifies how a client sends a request to a server through the ORB:



**Figure (4.11):** The structure of ORB interface

#### 4.6.1 Interface Definition Language (IDL).

A cornerstone of the CORBA standards is the Interface Definition Language. IDL is the OMG standard for defining language-neutral APIs and provides the platform-independent description of the interfaces of distributed objects. The ability of the CORBA environments to provide consistency between clients and servers in heterogeneous environments begins with a standardized definition of the data and operations constituting the client/server interface. This standardization mechanism is the IDL, and is used by CORBA to describe the interfaces of objects. IDL defines the modules, interfaces and operations for the applications and is not considered a programming language. The various programming languages, such as Ada, C++, or Java, supply the implementation of the interface via standardized IDL mappings.

#### 4.6.2 How is CORBA works

1. Create the IDL to Define the Application Interfaces: The IDL provides the operating system and programming language independent interfaces to all services and components that are linked

to the ORB. The IDL specifies a description of any services a server component exposes to the client. The term "IDL Compiler" is often used, but the IDL is actually translated into a programming language.

2. Translate the IDL: An IDL translator typically generates two cooperative parts for the client and server implementation, stub code and skeleton code. The stub code generated for the interface classes is associated with a client application and provides the user with a well-defined Application Programming Interface (API). In this example, the IDL is translated into C++.
3. Compile the Interface Files: Once the IDL is translated into the appropriate language, C++ in this example, these interface files are compiled and prepared for the object implementation.
4. Complete the Implementation: If the implementation classes are incomplete, the spec and header files and complete bodies and definitions need to be modified before passing through to be compiled. The output is a complete client/server implementation.
5. Compile the Implementation: Once the implementation class is complete, the client interfaces are ready to be used in the client application and can be immediately incorporated into the client process. This client process is responsible for obtaining an object reference to a specific object, allowing the client to make requests to that object in the form of a method call on its generated API.
6. Link the Application: Once all the object code from steps three and five have been compiled, the object implementation classes need to be linked to the C++ linker. Once linked to the ORB library, in this example, *ORBexpress*, two executable operations are created, one for the client and one for the server.
7. Run the Client and Server: The development process is now complete and the client will now communicate with the server. The server uses the object implementation classes allowing it to communicate with the objects created by the client requests.

In its simplest form, the server must perform the following:

- Create the required objects.



- Notify the CORBA environment that it is ready to receive client requests.
- Process client requests by dispatching the appropriate servant.

#### **4.6.3 CORBA Features**

- Language – neutral Data Type
- Local /Remote Transparency
- Dynamic Method Invocation
- High level Binding
- Self Describing Meta Data

#### **4.6.4 Encryption**

CORBA supports encryption client request messages as found by Rakman, Ch., *et al.* (2006). They found that “After completion of security association, the ORB/client calls the security association block again to encrypt the client's request message and sends the encrypted request message to the ORB/target. The ORB/target decrypts the client's request message using the security association block and calls the access control block to decide whether access to the target is allowed.”

#### **4.6.5 CORBA advantages**

Here we discuss some advantages of using CORBA in the design and implementation of network management systems and in other types of applications.

- Object Location Transparency: The client does not need to know where an object is physically located. An object can either be linked into the client, run in a different process on the same machine, or run in a server on the other side of the planet. A request invocation looks the same

regardless, and the location of an object can change over time without, breaking applications.

- **Server Transparency:** The client is, as far as the programming model is concerned, ignorant of the existence of servers. The client does not know (and cannot find out) which server hosts a particular object, and does not care whether the server is running at the time the client invokes a request.
- **Language Transparency:** Client and server can be written in different languages. This fact encapsulates the whole point of CORBA; that is, the strengths of different languages can be utilized to develop different aspects of a system, which can interoperate through IDL. A server can be implemented in a different language without clients being aware of this.
- **Implementation Transparency:** The client is unaware of how objects are implemented. A server can use ordinary flat files as its persistent store today and use an OO database tomorrow, without clients ever noticing a difference (other than performance).
- **Architecture Transparency:** The idiosyncrasies of CPU architectures are hidden from both clients and servers. A little-endian client can communicate with a big-endian server with different alignment restrictions.
- **Operating System Transparency:** Client and server are unaffected by each other's operating system. In addition, source code does not change if you need to port the source from one operating system to another
- **Protocol Transparency:** Clients and servers do not care about the data link and transport layer. They can communicate via token ring, Ethernet, wireless links, ATM (Asynchronous Transfer Mode), or any number of other networking technologies.
- **Creating an extensible application framework.** CORBA provides a good means for tackling the problem that comes from rapid change in network technology; as it is based on an object model. By using CORBA and object-oriented programming techniques, ProSphere software can be enhanced and maintained easily. The ProSphere architecture defines an abstract set of CORBA objects, which model all basic types of network objects. These objects are largely based on the TMN Generic Network

Model and define the network topology objects and resources that are shared by ProSphere applications.

- Providing an open/standard interface. With standard interfaces, the management system need not know vendor-specific information and can handle network resource management in a common way. Although it is not clear what technology will ultimately be used for such interfaces, there is strong support in the telecommunications industry to use CORBA for this purpose.
- Decoupling user interfaces. In the past, typical network management applications were developed so that the application code was coresident with the user interface code. By having a protocol-defined interface, CORBA forces the separation of an object interface and implementation from its use. The ProSphere user interfaces use the compiled stubs from IDL to interact with the objects. This decouples the user interface code completely from the object implementation.
- Web-based management. Support for Web-based management is a direct consequence of implementing the user interfaces in Java. Also, the native protocol used for communication in ProSphere is the Internet Inter-ORB Protocol (IIOP), implying that ProSphere can be accessed through a Web browser such as Netscape Navigator or Internet Explorer.
- Integration with leading commercial NMS products. By using CORBA we are able to decouple the integration with OpenView from the rest of ProSphere and thereby eliminate dependencies. It should be noted that integration with any other software product would again be as loosely coupled as the OpenView integration.

#### **4.6.6 CORBA Disadvantages**

Philip Maechling (2005) and Doland L. (2008) classified the disadvantages as follows:-

- Described services require the use of an interface definition language (IDL) which must be learned. Implementing or using services require and IDL mapping to your required language – writing one for language that isn't supported would take large amount of work.
- IDL to language mapping tools create code stubs based on the interface – some tools may not integrate new changes with existing code.
- CORBA does not support the transfer of object, or code.
- Not all classes of applications need real – time performance, and speed may be traded off against ease of use for Java system
- Communication overhead is very high with CORBA.
- Communication is done in bulk of at least 100,000 messages to minimize the effect of overhead.
- CORBA also requires a lot of initial setup, for even very basic communication
- CORBA is designed to use single server or a single pool of servers; consequently the behaviour of multiple servers interacting is not suited for computationally intensive operations.

## 4.7 OLAP

Is the short for **Online Analytical Processing**, a category of software tools that provides analysis of data stored in a database. OLAP tools enable users to analyze different dimensions of multidimensional data. For example, it provides time series and trend analysis views. OLAP often is used in data mining.

The term, On Line Analytical Processing, was coined in 1993 by Ted Codd who is referred as, the father of the relational database. Express was the first software that performed OLAP queries and was released in 1970 but later acquired by Oracle in 1995. But OLAP was not a mainstream product until 1998, when Microsoft, released its own OLAP server called, Microsoft Analysis Services.

“OLAP never had wide spread APIs which were enjoyed by other relational databases until 1997, when Microsoft, introduced the first real API, OLE DB for OLAP or ODBO and also introduced Multidimensional Expressions or MDX query language. MDX provides powerful syntax for manipulating the multi-dimensional data. However, huge data volumes are analyzed as numeric facts, distributed along descriptive dimensions (Svetlana Mansmann, 2008).”

“Online analytical processing or OLAP is an approach to quickly answer multi-dimensional analytical queries. OLAP is part of the broader category of business intelligence, which also encompasses relational reporting and data mining. The typical applications of OLAP are in business reporting for sales, marketing, management reporting, business process management (BPM), budgeting and forecasting, financial reporting and similar areas. The term OLAP was created as a slight modification of the traditional database term OLTP (Online Transaction Processing) (Wikipedia, 2008).”

OLAP can be used for data mining or the discovery of previously undetected relationships between data items. An OLAP database does not need to be as large as a data warehouse, since not all transactional data is needed for trend analysis. Using Open Database Connectivity (ODBC), data can be imported from existing relational databases to create a multidimensional database for OLAP.

Two leading OLAP products are Hyperion Solution's Essbase and Oracle's Express Server. OLAP products are typically designed for multiple-user environments. Moreover, the main component of OLAP is the OLAP server which is a high capacity, multi-user manipulation engine designed to operate multi-dimensional data and it sits between a client and database management systems or DBMS. An OLAP server is designed to support fast extemporized information in any alignment and also designed for rapid calculations.

### **4.7.1 OLAP Design**

Visual Studio Team System (2009) from Microsoft revealed that, OLAP databases provide aggregated summary information quickly using a schema that is easily understood by end users. The OLAP design makes it unnecessary to "join" tables together as in a relational database, and includes all of the relationships required to enable a specific set of queries to be run against the database. The central object in an OLAP database is called a cube. A single OLAP database may contain many cubes; however, the team system database consists of a single cube.

The following are the types of OLAP systems

1. MOLAP - Multidimensional Online Analytical Processing or MOLAP is considered to be main form of OLAP and thus sometimes just referred as OLAP. It is known for its fast query performance which is because of optimized storage, multidimensional indexing and caching. One of the advantage of MOLAP is that array model provides natural indexing. The main disadvantage is that it introduces data redundancy.
2. ROLAP - Relational Online Analytical Processing is an alternative to the MOLAP and works directly with relational databases. The main difference between MOLAP and ROLAP is that it doesn't require storage of information. One advantage of ROLAP is that data stored in database can be accessed through SQL tool. But it is considered to be slower than MOLAP tools by industry people.

### **4.7.2 OLAP Security**

For the communication between the front-end applications and the OLAP server (or the data warehouse in 2-tier environments) usually a client/server connection will be utilized, possibly to remote sites. Even though the information on this channel is most likely aggregated and less complete, it may be highly security critical. The use of the Internet or other possibly insecure networks for the above

mentioned connections makes suitable security measures necessary. As only some tools support encrypted communication on application level, virtual private network (VPN) technology might be appropriate (Torsten, P., and Günther, P., 2000).

### **4.7.3 OLAP Advantages**

The following are some of the benefits of OLAP:-

1. The main benefit of the OLAP is its steadiness in calculations. The reporting is always represented in a coherent presentation irrespective of how fast data is dealt with through the OLAP server or software and thus allows the executives and analysts to know exactly to look for where.
2. The other convenience of OLAP is that it allows the manager to tear down data from OLAP database in specific or broad terms.
3. OLAP doesn't suffer from the symmetry problem that limits SQL.

### **4.7.4 OLAP Disadvantages**

Ralph Kimball (2007) stated some disadvantages as follows:-

1. The big objection to OLAP is the proprietary, non-standard character of each vendor's OLAP offering.
2. Don't expect to port an OLAP implementation to another vendor's product. Discard everything including all your application development
3. There is no accepted, universally implemented access language for OLAP, although Microsoft's MDX is the closest thing to a standard access language. Significantly, Oracle has not embraced MDX, preferring to rely on SQL for all forms of database access.
4. MDX in its full glory may be too complex for IT personnel to write by hand, or understand a complex application.

5. OLAP cubes can be catastrophically invalidated if you are not careful, i.e. a Type 1 SCD change to a dimension may cause all the cubes using that dimension to rebuild!
6. OLAP cubes are not considered stable enough for serious archiving and backup: this is a strong reason for creating a set of dimensional relational tables duplicating the content of the cube for those purposes.
7. When you must rebuild a cube, it may be a very time consuming process.

## **4.8 OLE**

OLE DB is a set of COM-based interfaces that expose data from a variety of sources. OLE DB interfaces provide applications with uniform access to data stored in diverse information sources, or data stores. These interfaces support the amount of DBMS functionality appropriate to the data store, enabling the data store to share its data (Microsoft Corporation, 2009).

OLE is Microsoft's framework for a compound document technology. Briefly, a compound document is something like a display desktop that can contain visual and information objects of all kinds: text, calendars, animations, sound, motion video, 3-D, continually updated news, controls, and so forth. Each desktop object is an independent program entity that can interact with a user and also communicate with other objects on the desktop. Part of Microsoft's ActiveX a technology, OLE takes advantage and is part of a larger, more general concept, the Component Object Model (COM) and its distributed version, DCOM. An OLE object is necessarily also a component (or COM object).

In simple words, ole enables us to create objects with one application and then link or embed them in a second application. For example, a desk-top publishing system might send some text to a word processor or a picture to a bitmap editor using ole. Embedded objects retain their original format and links to the application that created them (wickedwillie, 2004).



### 4.8.1 OLE DB Programming Overview

What is OLE DB, and what makes it distinct from other database technologies? OLE DB is a high-performance, COM-based database technology produced by Microsoft. What sets OLE DB apart from Microsoft's other database technologies is how it provides Universal Data Access.

### 4.8.2 Universal Data Access

Universal Data Access provides a common way to access data regardless of the form in which it is stored. In a typical business situation, a vast amount of information is stored outside of corporate databases. This information is found in file systems (such as FAT or NTFS), indexed-sequential files, personal databases (such as Access), spreadsheets (such as Excel), project planning applications (such as Project), and e-mail (such as Outlook).

Accessing this data using the various associated applications presents a major bottleneck in workflow or at least an annoyance. Most companies find themselves in this situation and deal with the problem by consolidating the information in a database management system (DBMS). However, such a move is expensive, time consuming, and in many cases not practical. The alternative is to develop a Universal Data Access solution. OLE DB and ADO provide Universal Data Access capability. Of the two, OLE DB is the more performance intensive and is recommended for use with Visual C++ applications.

Universal Data Access implies two capabilities: the first is distributed query or uniform access to multiple (distributed) data sources and the second is the ability to make non-DBMS data sources accessible to database applications.

- i. Distributed query: Is the ability to access data uniformly on multiple data sources. The data sources can be of the same type (such as two separate Access databases) or of different types (such as a SQL Server database

and an Access database). Uniformly means that you can meaningfully run the same query on all data sources.

- ii. Non-DBMS access: The ability to make non-DBMS data sources accessible to database applications. Examples of DBMS data sources include IMS, DB2, Oracle, SQL Server, Access, and Paradox. Examples of non-DBMS data sources include information in file systems, e-mail, spreadsheets, and project management tools.

### **4.8.3 Benefits of COM**

This is where COM comes in. OLE DB is a set of COM interfaces. By accessing data through a uniform set of interfaces, you can organize a database into a matrix of cooperating components.

Based on the COM specification, OLE DB defines an extensible and maintainable collection of interfaces that factor and encapsulate consistent, reusable portions of DBMS functionality. These interfaces define the boundaries of DBMS components such as row containers, query processors, and transaction coordinators, which enable uniform transactional access to diverse information sources.

Typically, OLE DB applications are written as DLLs, but its COM implementation overcomes the disadvantages of DLLs (such as naming and version problems) by using componentized code. In OLE DB, you call interfaces or access other components using their globally unique identifiers (GUIDs).

Moreover, COM keeps track of component usage by using reference counting. When you call a method on an interface, the reference count is incremented; when the method returns, the reference count is decremented. When the count equals zero, the component to which the method belongs is released.

#### 4.8.4 Drawbacks of COM

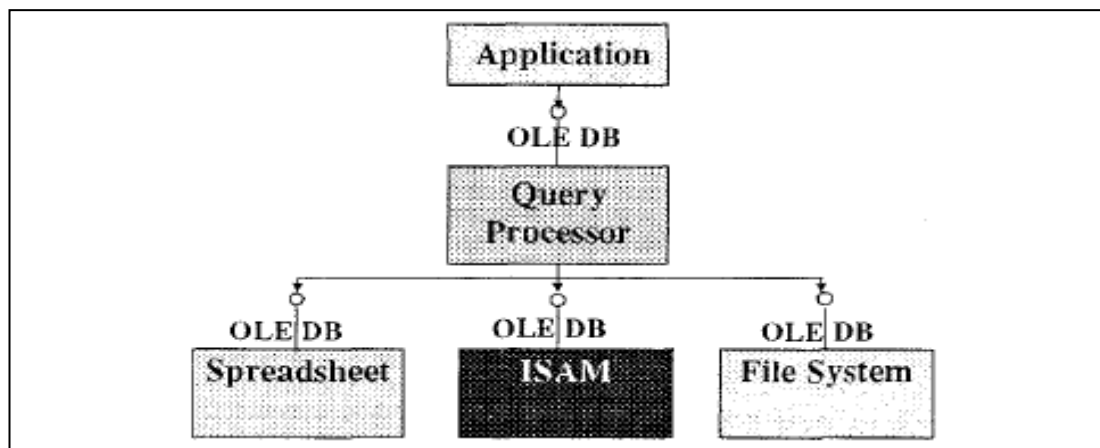
- While COM is available platform for enterprise-level Internet applications today, it also has limitations of its own for Internet development as described by Fred Barwell, *et al.* (2001a).
- COM-base applications are subject to major deployment and configuration issues. Small changes in COM interfaces can accidentally render entire applications inoperable.
- All of the COM-based components require registration of their class IDs on the local client machine. These GUID-based identifiers must be placed in the local client's Windows Registry. This is typically done by a complex installation program.
- Lack of interoperability with other platforms
- Lack of Built-In Inheritance

#### 4.8.5 OLE DB Architecture

In OLE DB, a client is any system or application component that consumes an OLE DB interface; this includes OLE DB components themselves. A provider is any component that exposes an OLE DE) interface (Blakeley, J.A., 1996).

#### 4.8.6 How OLE Works

JOS6A. Blakeley (1996) thinks that OLE DB lowers the entry bar for simple tabular data providers by requiring them to implement only the functionality native to their data store. At a minimum, a provider needs to implement the interfaces necessary to expose data as tables. This opens the opportunity for the development of query processor components (e.g., SQL, geographical) that can consume tabular information from, any provider that exposes its data through the OLE DB API as shown in the figure (4.12) below.



**Figure (4.12):** A component DBMS architecture using OLE DB interfaces.

#### 4.8.7 Oracle Objects for OLE (OO4O)

The Oracle Objects for OLE (OO4O) were introduced to capitalize on the huge success of the Microsoft COM (Component Object Model) standard. Implementing OO4O allowed COM-compliant applications to connect to the Oracle RDBMS directly, bypassing ODBC, and thus increasing efficiency and raw speed of the applications.

The OO4Os consist of an in-process OLE automation server (DLL), which provides an OLE/ActiveX interface to COM-compliant applications, specifically Visual Basic and Visual Basic for Application (VBA); OCX custom Data Control, to facilitate data manipulations; and two C++ class libraries — one for Microsoft Foundation Classes (MFC) specification and one for Borland (OWL).

#### 4.8.8 OLE Advantages

Dr. Bruce E. and Kevin S. (2008) revealed that the OLE advantages can be classified into linking an embedding and OLE Automation and detailed as follows:-

- **Linking and Embedding:** Linking and embedding is a common means of inserting objects into Word documents, Excel worksheets, and Power Point slides. Likewise, it can be used to insert objects into your own Windows application. In this manner, your Windows application becomes the host document. An object is embedded in a host document if the document contains the object. That is, the object can be activated, edited, and saved within the document containing the object. If an object is linked in a host document, the document contains a reference to the data and image of the linked object but does not store the data (Hergert, 1995). Thus the document contains a view of the object's data which is stored in an external file.
- **OLE Automation:** In OLE automation, an OLE server provides objects, properties, and methods for OLE clients to use at run time. By exposing its objects in this manner, OLE servers allow OLE clients to take complete control of the objects. Since these exposed objects are not in the clients code space, any number of clients can employ the servers objects at the same time. Both Visual Basic and Visual C++ are suited well for OLE automation programming. While both allow developers to browse OLE server objects for available properties, methods, and events, Visual Basic makes it relatively quick and easy to access and control exposed objects.

#### **4.8.9 OLE disadvantages**

Clifton Karnes (1994) shows that "We've seen a lot of the benefits of OLE, but there's a downside, too. There are three primary disadvantages to OLE: broken links, large files, and big memory requirements. Broken links occur when you move a linked file. If this happens, most containers have a way for you to manually repair the link, but it's a pain. The next version of OLE should do a better job of tracking links. If you embed objects your files can quickly become huge--easily several megabytes. Not only do these files take up lots of disk space, but they eat up RAM as well when you're working with them. And speaking of RAM, when you edit an OLE object and run its server, both container and server must be running at the same time.

With programs the size of Word and Excel, I'd consider 8MB of memory a minimum to accomplish this.”

However, we can summarize the OLE benefits as follows:-

1. Edit embedded objects in place (this is called visual editing).
2. Drag and drop objects between documents.
3. Control other OLE objects with OLE automation.
4. OLE also allows people with differing levels of expertise to participate in the development of these Windows applications.
5. OLE; DB leverages the COM infrastructure, which reduces unnecessary duplication of services and provides a higher degree of interoperability not only among diverse information sources, but also among programming environments and tools already developed for this environment.
6. Distributed OLE DB offers an infrastructure for clients to interact with providers transparently across process and machine boundaries. OLE DB client code is entirely unaware of whether the provider component is executing locally or remotely.
7. OLE provides a unified, expandable, object-oriented communication protocol for the Windows 32-bit platform (Kruglinski, 1996).

## **4.9 XML**

XML is a markup language for structured documentation. Structured documents are documents that contain both content (words, pictures, etc.) and some indication of what role that content plays (for example, content in a section heading has a different meaning from content in a footnote, which means something different than content in a figure caption, etc.). Almost all documents have some structure.

Works by Jessica Heasley (2004) stated that XML is not exactly a language, but more of a standard for creating documents that meet XML criteria. In other words, XML describes the syntax, which is better for data exchange. XML files include tags as shown in the following example:

```
<name>
  <first>Jane</first>
  <last>Doe</last>
</name>
```

This XML data structure is being used to hold data about a name that could be shared with other programs. The XML data and required formatting is of course lengthier than the plain text version of the name, but it is easier to write software to access the specific last name when the data is given a structure or tag.

## **4.9.1 Features / Capabilities**

### **4.9.1.1 Oracle**

The growth in the use of XML has created a requirement to manage XML with the same degree of rigor as is required for other mission-critical data. To meet this need, Oracle developed Oracle XML DB. Oracle XML DB is a high-performance, native XML storage and retrieval technology that is delivered as a part of all versions of Oracle Database. Oracle XML DB provides full support for all of the key XML standards, including XML, Namespaces, DOM, XQuery, SQL/XML and XSLT. The major goal of Oracle Database 11g was to deliver increased flexibility combined with improved performance. This was achieved by further optimization of XML Schema-based storage and the introduction of additional storage and indexing techniques for non-schema-based XML content (Mark Drake, 2007).

#### **4.9.1.2 Microsoft SQL Server 2005**

Shankar Pal, *et al.* (2005) published that, eXtensible Markup Language (XML) has been widely adopted as a platform-independent format for data representation. It is useful for exchanging information among loosely coupled, disparate systems, such as in business-to-business (B2B) applications and workflow situations. Data interchange has been a major driver of XML technologies.

XML is increasingly present in enterprise applications that are used for modeling semi-structured and unstructured data. One such application is document management. Documents (e-mail messages, for example) are semi-structured by nature. If documents are stored inside a database server as XML, powerful applications can be developed such as:

- Applications that retrieve documents based on their content.
- Applications that query for partial content, such as finding the section whose title contains the word "Background."
- Applications that aggregate documents.

#### **4.9.2 How XML Schema Processing with SQL server 2005?**

XML columns, variables, and parameters can optionally be typed according to a collection of XML schemas that may be related (for example, by using <xs:import>) or unrelated to one another. Each typed XML instance specifies the target namespace from the XML schema collection it conforms to. During data assignment and modification, the database engine validates the instance according to the XML schema.



### 4.9.3 Encryption

Nowadays, XML has been used broadly in open distributed networks, such as e-commerce applications for Internet. An environment such as the Internet cannot be considered a safe place. Thus, multi-agent systems should have security mechanisms, e.g. confidentiality and integrity. The XML Security Specifications are standards that are based on XML and provide security mechanisms. They include: XML Digital Signature for digital signature; XML Encryption for cryptography.

XML document security is developed by several organizations, including the World Wide Web Consortium (W3C), the Organization for the Advancement of Structured Information Standards (OASIS), and some related work by the Internet Engineering Task Force (IETF). However, the XML Encryption recommendation from the W3C defines mechanisms for encrypting XML documents to ensure confidentiality. The specification defines a range of cryptographic mechanisms, including the use of both conventional and public key cryptography. It provides a system of fine-grained control of encryption, so that only certain parts of a document may be encrypted (Win Treese, 2002).

According to studies by Emerson, O., *et al.* (2006), the XML security standards and specifications include:

- XML Signature: A standard specification developed jointly by W3C and IETF. An XML signature is equivalent to a digital signature; it can be used to digitally sign XML documents, like SOAP messages.
- XML Encryption: A standard specification developed by W3C that proposes to encrypt XML documents. This specification can be used to assure confidentiality in case of a security context ranging over several SOAP intermediaries.
- XML Key Management Specification (XKMS): Developed by W3C to allow clients to obtain cryptographic key information (such as keys and certificates). It also describes protocols for key management, such as registration and revocation.

In the same way, the XML Digital Signature recommendation from the W3C defines a set of mechanisms for digital signatures on XML documents. Digital signatures provide both authentications of the sending party as well as integrity protection to ensure that messages have not been modified, whether maliciously or accidentally. Digital signatures may be attached as part of a document, or they may be stored and communicated separately (Win Treese, 2002).

Works by Ernesto, D. *et al.* (2002) have shown that “The application to XML data of the latest advancement of public key cryptography has remedied most of the security problems in communication; commercial products are becoming available (such as AlphaWorks’ XML Security Suite) providing fine-grained security features such as digital signatures and element wise encryption to transactions involving XML data as a way to meet authenticity, secrecy, and nonrepudiation requirements in XML-based transactions.”

Based on the analysis of related works, they propose an XML technology-based solution WALSG to web security. The basic idea of WALSG is to check and verify the parameters of web forms, cookies of a web site, and access directories of a web site according to the security policies and access control policies specified by administrators, and apply XML Signature and XML Encryption when necessary. WALSG is resided between firewalls and web servers and automatically protects the whole web applications on web servers. As a URL request is transferred from clients to servers, WALSG automatically allows or denies the request according to the corresponding security policies and access control policies defined for each HTML page (Teng, L., and Ping, Y., 2006).

#### **4.9.4 XML Advantages:-**

- The first benefit of XML is that because we are writing our own markup language, we are not restricted to a limited set of tags defined by proprietary vendors. Rather than waiting for standards bodies to adopt tag

set enhancements (a process which can take quite some time), or for browser companies to adopt each other's standards, with XML, we can create our own set of tags at our own pace.

- We are free to develop tools that meet our needs exactly.
- By defining our own tags, we create the markup language in terms of our specific problem set! Rather than relying on a generic set of tags which suits everyone's needs adequately, XML allows every person/organization to build their own tag library which suits their needs perfectly.
- XML allows each specific industry to develop its own tag sets to meet its unique needs without forcing everyone's browser to incorporate the functionality of huge amount of tag sets, and without forcing the developers to settle for a generic tag set that is too generic to be useful.
- The real power of XML comes from the fact that with XML, not only can you define your own set of tags, but the rules specified by those tags need not be limited to formatting rules. XML allows you to define all sorts of tags with all sorts of rules, such as tags representing business rules or tags representing data description or data relationships.
- XML uses the Common Component Architecture or CCA, and the Common Object Request Broker Architecture, or CORBA. In other words, this means that it uses a common and standard protocol which helps interoperability for programs. It also allows RMI, or remote method invocation in Java and invokes another java object. It also allows the clients to connect to the program using the remote procedure calling, or RPC in short.
- Standard: They were well standardized through a lot of examinations by the W3C community and the several implementations have been already developed.
- XML is easy to convert further into different formats as required: e.g HTML, PDF, and plain text. This gives flexibility to web applications where data can be searched for and accessed from the database, and then formatted for output in different formats using e.g XSL.

- XML is already a standard for data interchange, so you may need to pass your data on to others as XML or take XML as input. There are advantages over competing technologies such as EDI. A feature of XML allows XML from different sources to be processed together to give a combined result.
- In Web applications, XML can theoretically be used to reduce server hits and load on the server for sorts etc, because some processing can be done by the client browser. However, in practice many browsers in use have no or limited XML capability.
- Storing XML data in a relational database provides benefits in the areas of data management and query processing. Microsoft SQL Server and Oracle XML DB provide powerful query and data modification capabilities over relational data. In SQL Server 2005, these capabilities are extended to querying and modifying XML data. This allows your company to leverage investments made over past releases, such as investments in the areas of cost-based optimizations and data storage. For example, indexing techniques in relational databases are well-known. These have been extended to indexing XML data so that queries can be optimized using cost-based decisions.
- XML data can interoperate with existing relational data and SQL applications. This means that XML can be introduced into the system as data modeling needs arise without disrupting existing applications. The database server also provides administrative functionality for managing XML data (for example, backup, recovery, and replication).
- Native XML support within SQL Server 2005 is necessary to address increasing XML usage. Enterprise application development benefits from the XML support in SQL Server 2005.

#### **4.9.5 XML Disadvantages**

- The extensive markup language is the way to go for developing future web applications, and it almost defines the future of web development.

There are no doubts about its performance in this arena. However, XML also has some draw backs which need to be looked at and improved upon. The reason it faces some resistance from users is a result of these drawbacks. One of the biggest drawbacks of XML is that it is lacking in the area of adequate applications for processing.

- There are no XML browsers on the market yet (although the latest version of IE does a pretty good job of incorporating XSL and XML documents provided HTML is the output).
- Thus, XML documents must either be converted into HTML before distribution or converting it to HTML on-the-fly by middleware. Barring translation, developers must code their own processing applications (Selena So, 2004).
- Oracle XML DB offers a XML structured storage. The main disadvantage of using structured storage is its lack of flexibility as only documents that match the XML schema can be stored.

## **4.10 Windows.NET**

### **4.10.1 What is the .NET Framework and Visual Studio?**

The .NET Framework is a development and execution environment that allows different programming languages and libraries to work together seamlessly to create Windows, Web, Mobile, and Office applications that are easier to build, manage, deploy, and integrate with other networked systems or as stand-alone applications. Visual Studio is the Integrated Development Environment (IDE) in which developers work when creating programs in one of many languages, including Visual Basic, for the .NET Framework. Visual Basic 2008 Express Edition is a no-cost, streamlined, easy-to-use development tool (msdn, 2009).

The Microsoft .NET Framework is a software framework available with several Microsoft Windows operating systems. It includes a large library of coded solutions to prevent common programming problems and a virtual machine that

manages the execution of programs written specifically for the framework. The .NET Framework is a key Microsoft offering and is intended to be used by most new applications created for the Windows platform.

The framework's Base Class Library provides a large range of features including user interface, data and data access, database connectivity, cryptography, web application development, numeric algorithms, and network communications. The class library is used by programmers, who combine it with their own code to produce applications.

Programs written for the .NET Framework execute in a software environment that manages the program's runtime requirements. Also part of the .NET Framework, this runtime environment is known as the Common Language Runtime (CLR). The CLR provides the appearance of an application virtual machine so that programmers need not consider the capabilities of the specific CPU that will execute the program. The CLR also provides other important services such as security, memory management, and exception handling. The class library and the CLR together compose the .NET Framework (Wikipedia, 2009).

#### **4.10.2 General Goals of .NET**

According to Barwell, *et al.* (2001b), many of the goals Microsoft had in mind when designing .NET reflect the limitations we previously discussed for development with previous tools and technologies. These goals include:

- Generation of highly distributed applications.
- Simplified software development.
- Better user interface over the Web.
- Simplified Deployment.
- Support for a variety of languages.
- An extendable platform for the future

- Future portability of compiled applications

#### 4.10.3 .NET Architecture

- Common Language Infrastructure (CLI): The core aspects of the .NET Framework lie within the Common Language Infrastructure, or CLI. The purpose of the CLI is to provide a language-neutral platform for application development and execution, including functions for exception handling, garbage collection, security, and interoperability, see figure (13).
- Assemblies: The intermediate CIL code is housed in .NET assemblies. As mandated by specification, assemblies are stored in the Portable Executable (PE) format, common on the Windows platform for all DLL and EXE files. The assembly consists of one or more files, one of which must contain the manifest, which has the metadata for the assembly. The complete name of an assembly (not to be confused with the filename on disk) contains its simple text name, version number, culture, and public key token. The public key token is a unique hash generated when the assembly is compiled, thus two assemblies with the same public key token are guaranteed to be identical from the point of view of the framework. A private key can also be specified known only to the creator of the assembly and can be used for strong naming and to guarantee that the assembly is from the same author when a new version of the assembly is compiled (required adding an assembly to the Global Assembly Cache).
- Metadata: All CLI is self-describing through .NET metadata. The CLR checks the metadata to ensure that the correct method is called. Metadata is usually generated by language compilers but developers can create their own metadata through custom attributes. Metadata contains information about the assembly, and is also used to implement the reflective programming capabilities of .NET Framework.
- Security: .NET has its own security mechanism with two general features: Code Access Security (CAS), and validation and verification. Code

Access Security is based on evidence that is associated with a specific assembly. Typically the evidence is the source of the assembly (whether it is installed on the local machine or has been downloaded from the intranet or Internet). Code Access Security uses evidence to determine the permissions granted to the code. Other code can demand that calling code is granted a specified permission. The demand causes the CLR to perform a call stack walk: every assembly of each method in the call stack is checked for the required permission; if any assembly is not granted the permission a security exception is thrown.

- **Class library:** The .NET Framework includes a set of standard class libraries. The class library is organized in a hierarchy of namespaces. Most of the built in APIs are part of either System.\* or Microsoft.\* namespaces. These class libraries implement a large number of common functions, such as file reading and writing, graphic rendering, database interaction, and XML document manipulation, among others. The .NET class libraries are available to all .NET languages. The .NET Framework class library is divided into two parts: the Base Class Library and the Framework Class Library, see figure (14).
- **Memory management:** The .NET Framework CLR frees the developer from the burden of managing memory (allocating and freeing up when done); instead it does the memory management itself. To this end, the memory allocated to instantiations of .NET types (objects) is done contiguously from the managed heap, a pool of memory managed by the CLR. As long as there exists a reference to an object, which might be either a direct reference to an object or via a graph of objects, the object is considered to be in use by the CLR. When there is no reference to an object, and it cannot be reached or used, it becomes garbage. However, it still holds on to the memory allocated to it. .NET Framework includes a garbage collector which runs periodically, on a separate thread from the application's thread, that enumerates all the unusable objects and reclaims the memory allocated to them.



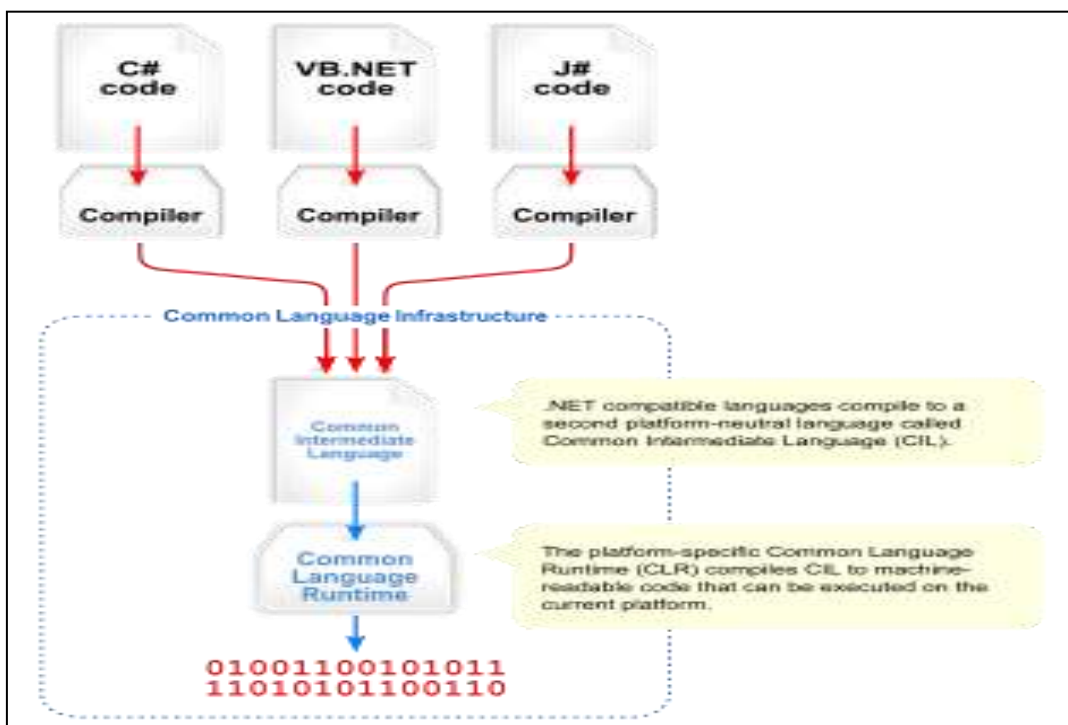


Figure (13): Visual Studio overview of the Common Language Infrastructure

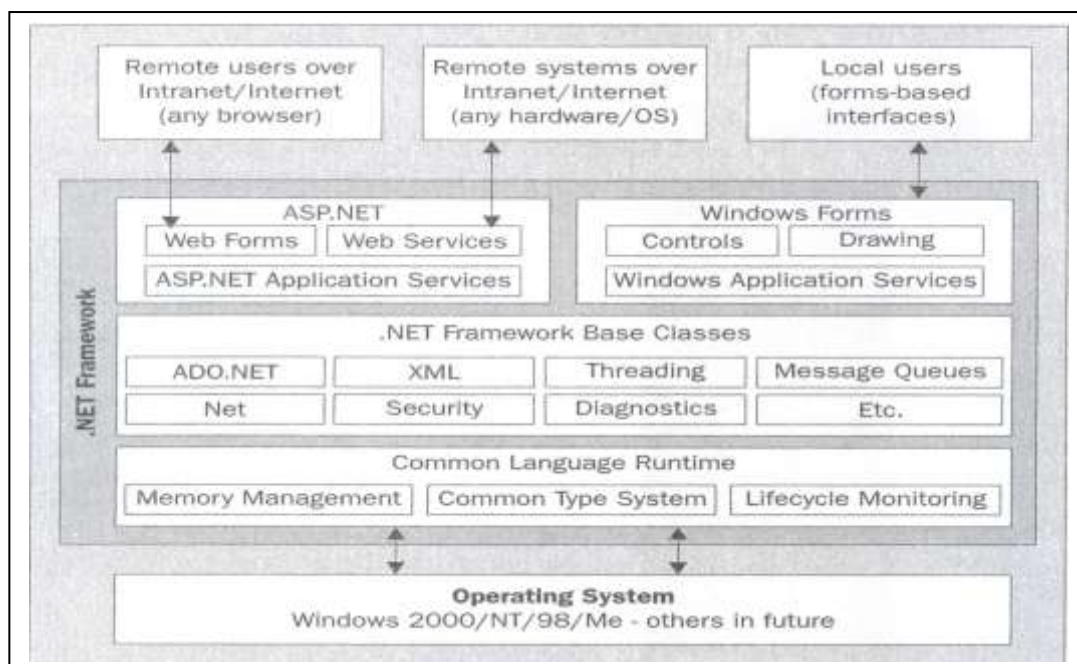


Figure (14): The Structure of Microsoft.Net

## 4.10.4 Principal Design Features

### 4.10.4.1 General features

- **Interoperability:** The interaction between new and older applications is commonly required; the .NET Framework provides means to access functionality that is implemented in programs that execute outside the .NET environment. Access to COM components is provided in the `System.Runtime.InteropServices` and `System.EnterpriseServices` namespaces of the framework.
- **Common Runtime Engine:** The Common Language Runtime (CLR) is the virtual machine component of the .NET framework. All .NET programs execute under the supervision of the CLR, guaranteeing certain properties and behaviours in the areas of memory management, security, and exception handling.
- **Language Independence:** The .NET Framework introduces a Common Type System, or CTS. The CTS specification defines all possible datatypes and programming constructs supported by the CLR and how they may or may not interact with each other. Because of this feature, the .NET Framework supports the exchange of instances of types between programs written in any of the .NET languages.
- **Base Class Library:** The Base Class Library (BCL), part of the Framework Class Library (FCL), is a library of functionality available to all languages using the .NET Framework. The BCL provides classes which encapsulate a number of common functions, including file reading and writing, graphic rendering, database interaction and XML document manipulation.
- **Simplified Deployment:** The .NET framework includes design features and tools that help manage the installation of computer software to ensure that it does not interfere with previously installed software, and that it conforms to security requirements.
- **Security:** The design is meant to address some of the vulnerabilities, such as buffer overflows, that have been exploited by malicious software.

Additionally, .NET provides a common security model for all applications.

- **Portability:** The design of the .NET Framework allows it to theoretically be platform agnostic, and thus cross-platform compatible. That is, a program written to use the framework should run without change on any type of system for which the framework is implemented. Microsoft's commercial implementations of the framework cover Windows, Windows CE, and the Xbox 360. In addition, Microsoft submits the specifications for the Common Language Infrastructure (which includes the core class libraries, Common Type System, and the Common Intermediate Language).

#### **4.10.4.2 ADO.NET Entity Framework**

The ADO.NET Entity Framework is designed to enable developers to create data access applications by programming against a conceptual application model instead of programming directly against a relational storage schema. The goal is to decrease the amount of code and maintenance required for data-oriented applications. Entity Framework applications provide the following benefits:

- Applications can work in terms of a more application-centric conceptual model, including types with inheritance, complex members, and relationships.
- Applications are freed from hard-coded dependencies on a particular data engine or storage schema.
- Mappings between the conceptual model and the storage-specific schema can change without changing the application code.
- Developers can work with a consistent application object model that can be mapped to various storage schemas, possibly implemented in different database management systems.
- Multiple conceptual models can be mapped to a single storage schema.
- Language-integrated query (LINQ) support provides compile-time syntax validation for queries against a conceptual model.

#### 4.10.4.3 What's New in the .NET Framework Version 3.5 SP1

1. ASP.NET: New ASP.NET features include ASP.NET Dynamic Data, which provides a rich scaffolding framework that allows rapid data driven development without writing code, and an addition to ASP.NET AJAX that provides support for managing browser history (back button support).
2. Common Language Runtime :Core improvements to the common language runtime include the following:
  - Improved application startup and working set performance.
  - Better layout of .NET Framework native images.
  - Opting out of strong-name verification of fully trusted assemblies.
  - Better generated code that improves end-to-end application execution time.
  - Detecting approaching full garbage collections with Garbage Collection Notifications.
  - Opting for managed code to run in ASLR (Address Space Layout Randomization) if supported by the operating system.
  - Managed applications that are opened from network shares have the same behavior as native applications by running with full trust.
3. .NET Framework Client Profile: The .NET Framework Client Profile is a subset of the full .NET Framework that targets client applications. This improves the installation experience on computers that do not already have the .NET Framework installed.
4. Windows Presentation Foundation: Performance improvements have been made to Windows Presentation Foundation, including a faster startup time and improved performance for Bitmap effects. Additional functionality for Windows Presentation Foundation includes better support for line of business applications, native splash screen support, DirectX pixel shader support, and the new Web Browser control.
5. Click Once: Click Once application publishers can decide to opt out of signing and hashing as appropriate for their scenarios, developers can programmatically install Click Once applications that display a

customized branding, and Click Once error dialog boxes support links to application-specific support sites on the Web.

6. Accessing Data with ADO.NET: The .NET Framework Data Provider for SQL Server (System.Data.SqlClient) provides full support for all the new features of the SQL Server 2008 Database Engine. For more information about.
7. Windows Communication Foundation: The Windows Communication Foundation (WCF) now makes the DataContract Serializer easier to use by providing improved interoperability support, enhancing the debugging experience in partial trust scenarios, and extending syndication protocol support for wider usage in Web 2.0 applications.
8. Windows Forms Controls: The Microsoft.VisualBasic.PowerPacks namespace introduces the new DataRepeater control that displays data in a customizable list format. This namespace also includes new vector shapes.

#### **4.10.5 Data Encryption in SQL Server (ADO.NET)**

Msdn (2009b) revealed that SQL Server 2005 provides functions to encrypt and decrypt data using a certificate, asymmetric key, or symmetric key. It manages all of these in an internal certificate store. The store uses an encryption hierarchy that secures certificates and keys at one level with the layer above it in the hierarchy. This feature area of SQL Server 2005 is called Secret Storage.

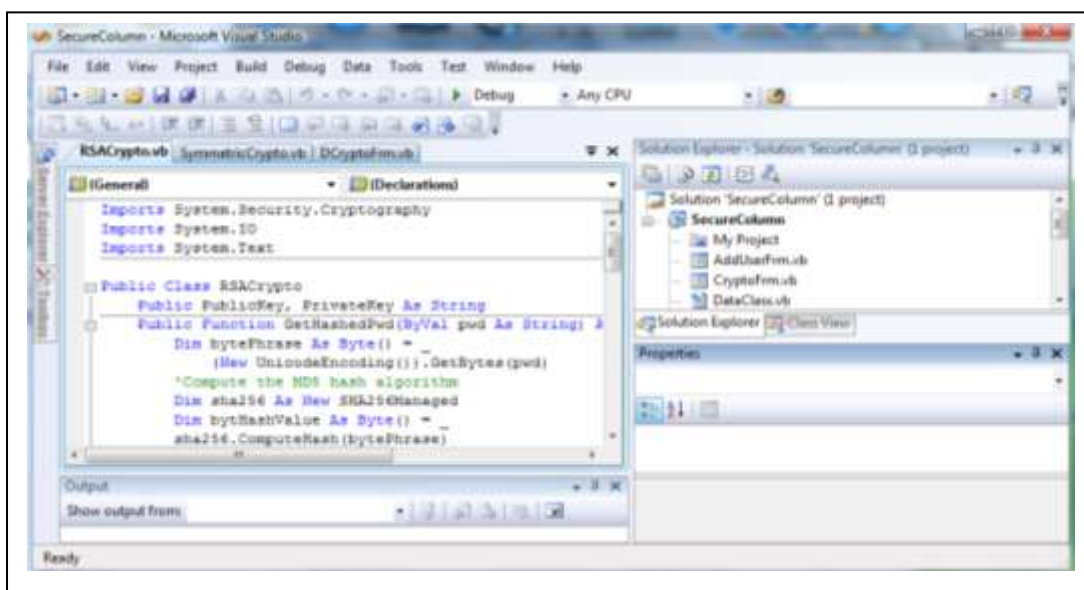
The fastest mode of encryption supported by the encryption functions is symmetric key encryption. This mode is suitable for handling large volumes of data. The symmetric keys can be encrypted by certificates, passwords or other symmetric keys.

- Keys and Algorithms: SQL Server 2005 supports several symmetric key encryption algorithms, including DES, Triple DES, RC2, RC4, 128-bit RC4, DESX, 128-bit AES, 192-bit AES, and 256-bit AES. The algorithms are implemented using the Windows Crypto API.

Within the scope of a database connection, SQL Server 2005 can maintain multiple open symmetric keys. An open key is retrieved from the store and is available for decrypting data. When a piece of data is decrypted, there is no need to specify the symmetric key to use. Each encrypted value contains the key identifier (key GUID) of the key used to encrypt it. The engine matches the encrypted byte stream to an open symmetric key, if the correct key has been decrypted and is open. This key is then used to perform decryption and return the data. If the correct key is not open, NULL is returned.

#### 4.10.6 Visual Basic.NET

Visual Basic (VB), formerly called Visual Basic .NET (VB.NET), is an object-oriented computer language that can be viewed as an evolution of Microsoft's Visual Basic (VB) implemented on the Microsoft Visual Studio figure (15). Its introduction has been controversial, as significant changes were made that broke backward compatibility with older versions and caused a rift within the developer community. The VB.NET has the following characteristics:



**Figure (15):** Microsoft Visual Studio (VB.NET)

#### 4.10.6.1 Simple

1. Design Windows Presentation Foundation (WPF) applications with built-in designer support
2. Create data-enabled applications with the lightweight SQL Server Compact Edition or powerful client/server applications with SQL Server 2008 Express
  - Build applications using LINQ (Language Integrated Query) which adds data querying capabilities for SQL Server, XML, and objects to Visual Basic
  - Support for the Entity Framework and designer tool
3. Develop quickly with IntelliSense support for the latest Visual Basic enhancements
4. Includes the Visual Basic Power Pack Line, Shape and Data Repeater controls

#### 4.10.6.2 Funny

1. Add cool, fun controls to your projects using the C4F Developer Toolkit and the C4F Vista P2P Developer Kit
  - Build cool applications with easy-to-use drag and drop controls which includes features like Smart Tags, all source code in Visual Basic, Quick Start documentation and much more; exclusively developed for Visual Studio 2008 and Windows Vista
  - Get started with the new WPF C4F Dashboard application
2. Share projects with others using the Popfly Explorer
3. Connect with your friends and create cool applications with the Facebook Developer Toolkit

### 4.10.6.3 Easy to Learn

1. Get started quickly by becoming familiar with the Visual Basic development environment with the Introduction to Visual Basic 2008 video
2. New to programming or Visual Basic? Start with the Beginner Developer Learning Center
3. Access the online documentation including samples and walkthroughs
4. Get online with your peers and connect with other beginners and hobbyists
  - Read the Visual Basic team's blogs
  - Get answers to your questions on the Visual Basic Express forum

### 4.10.7 Advantages of Using Managed Code to Create Database Objects

You can use .NET Framework languages in addition to the Transact-SQL programming language to create database objects and retrieve and update data for Microsoft SQL Server 2005 databases. Using Visual Basic, Visual C#, or Visual C++ projects, you can create stored procedures, triggers, aggregates, user-defined functions, and user-defined types. The following list is a summary of the advantages of using a .NET Framework language rather than Transact-SQL:

- Enhanced programming model .NET Framework languages offer constructs and capabilities previously unavailable to SQL developers.
- Enhanced Safety and Security Managed code runs in a common language runtime environment hosted by the database engine. This allows .NET Framework database objects to be safer and more secure than the extended stored procedures available in earlier versions of SQL Server.
- User-Defined Types and Aggregates User-defined types and user-defined aggregates are two new managed database objects which expand the storage and querying capabilities of SQL Server.



- **Common Development Environment** Database development is integrated into the Microsoft Visual Studio development environment. Developers use the same tools for developing and debugging database objects and scripts as they use to write middle-tier or client-tier .NET Framework components and services.
- **Better Performance** Some functions, such as those that run mathematical operations on every row in a database, might perform better when they are compiled assemblies that are built from a Visual Basic, Visual C#, or Visual C++ project rather than when they are written in Transact-SQL, which is interpreted code. For example, performance improvements will be achieved for functions, particularly those that perform integer operations. However, stored procedures that only access data will not perform better.
- **Language Richness** Visual Basic, Visual C#, and Visual C++ provide capabilities that are not available in Transact-SQL, such as arrays, sophisticated exception handling, and reusability of code.
- **Reusability of Code** A library of managed assemblies can be created and distributed more easily than a Transact-SQL script can be distributed.
- **Extensibility** Using Visual Basic, Visual C#, or Visual C++, you can create two database objects that cannot be created using Transact-SQL: aggregates and user-defined types.
- **Leverage Existing Skills** You can use and enhance your skills in the languages and development environment in which you are already experienced to create database objects.
- **Richer developer experience** When you develop database objects using the SQL Server project template, you have complete integration with the project system, including building, debugging, and deployment to multiple servers.
- **Stability and reliability** The database objects that you create using Visual Basic, Visual C#, or Visual C++ are more secure, stable, robust, and reliable than extended stored procedures, which might produce memory leaks or other problems that reduce the performance and reliability of the server. When you run stored procedures that were created using Visual Basic, Visual C#, or Visual C++, memory management and threading are not performed by the stored procedure and are therefore handled more robustly.

- **Security** When you use database objects created using Visual Basic, Visual C#, or Visual C++, the code-access security of those languages is combined with the user-based permissions in SQL Server.
- **Stored Procedures and Triggers:** Stored procedures are a precompiled collection of programming statements that perform operations in the database, and are stored under a name and processed as a unit.
- A trigger is a special kind of stored procedure that is activated when you modify data in a specified table using one or more of the data modification operations: update, insert, or delete. For more information about triggers, see the sql server documentation.
- Development of stored procedures and triggers is enhanced by the language richness of visual basic, visual c#, and visual c++, particularly when you are implementing the complex procedural logic that is required to enforce business rules. In addition, the .net framework contains many libraries. Of particular interest are those that provide the ability to manage many aspects of cryptography, the extensive math libraries, and the external access to web services, files, and business-to-business communication systems.
- **Functions:** Functions operate on one or more values to return either a scalar value or a table. For more information about the types of functions that the Transact-SQL programming language provides, see the SQL Server documentation.
- **Aggregates:** Aggregate functions are used to summarize all the data in a table. They perform a calculation on a set of values and return a single scalar value. For more information about the aggregate functions that are provided by the Transact-SQL programming language, see the SQL Server documentation.
- To supplement those aggregate functions, you can define new aggregates that perform more complex arithmetic functions. For example, you can perform a calculation on the data in many rows and return one value or create a concatenated string.
- **User-Defined Types:** Types specify the nature of data. For information about the set of system data types supplied with SQL Server, see the SQL Server documentation.
- Using Visual Basic, Visual C#, and Visual C++, you can define new types so that you are no longer limited to the predefined types that are supplied

with SQL Server. You can create simple types such as postal codes or more complex types for parsing the information returned from a credit card transaction. Also, when you are working with user-defined types, data can be interpreted and manipulated on both the SQL client and SQL Server; by using ADO.NET, you can download an assembly that contains a type definition from the SQL Server and use it to examine data on the SQL client.

- Faster development (less to do, the system handles more).
- More reuse because of inheritance.
- Higher scalability – many capabilities to help applications scale are built in .NET.
- Easier to build sophisticated development tools – debuggers and profilers can target the Common Language Runtime, and thus became accessible to all .NET-enabled language.
- Fewer bugs – whole classes of bugs should be unknown in .NET. With the CLR handling memory management, memory leaks should be a thing of the past, for example.
- Potentially better performance – Microsoft's heavy investment in system level code for memory management, garbage collection, and the like have yielded an architecture that should meet or exceed performance of typical COM-based applications today.

#### **4.10.8 Disadvantages of .NET**

- Applications running in a managed environment tend to require more system resources than similar applications that access machine resources more directly.
- As just-in-time compilation (JIT) languages can be more easily reverse-engineered than native code to algorithms used by an application, there is concern over possible loss of trade secrets and the bypassing of license control mechanisms. Many obfuscation techniques already developed can help to prevent this.
- In a managed environment the regularly occurring garbage collection for reclaiming memory suspends execution of the application for an

unpredictable lapse of time (typically no more than a few milliseconds, but in memory-constrained systems can be much longer). This makes such environments unsuitable for some applications such as those that must respond to events with predictable timing.

- Since the framework is not pre-installed on older versions of Windows an application that requires it must verify that it is present, and if it is not, guide the user to install it. This requirement may deter some from using the application as the downloads are many megabytes in size.

#### 4.11 Comparative Studies

In this section, some comparisons were made between some Database Access Technologies and their capabilities that facilitate usage. As mentioned earlier in Chapters 2 and chapter 4, there are many other technologies. Nevertheless, this comparison concentrates on nine only as detailed below. This selection is based on their common global application, and most of them have built-in on the databases. The first comparison is done in terms of supporting internet, network and database encryption (see Table 4.2).

**Table 4.2:** Comparison between DATs Capabilities

User Interface Technologies		Internet	Network	Database Encrypt/Decrypt
Extensible Markup Language (XML)		X	X	X
Java Database Connectivity (JDBC)	Type1 Driver			
	Type2 Driver			
	Type3 Driver	X	X	
	Type4 Driver	X	X	X
CORBA		X	X	
Object Linking and Embedding (OLE)			X	
Open Database Connectivity (ODBC)		X	X	X
ActiveX data objects (ADO)		X		

Online Analysis process (OLAP)		X	
ADO.NET	X	X	X
Microsoft.NET	X	X	X

Notes:- X = Applicable & Empty means not found my review.

As shown in the table 4.2, the strongest support to comparative parameter comes from XML, JDBC Type4 driver, ODBC, DOT.NET and Microsoft.NET. While, CORBA and JDBC Type3 driver come next. However, OLE and OLAO show no support to comparative criteria's.

The next contrast depends on prior comparative consequence. However, the best five DATs are subject to assessment of pros, cons and others as shown in table 4.3 below. All DATs show some advantages/advantages, but JDBC and Microsoft.Net demonstrate comprehensives pros and minor disadvantages.

**Table 4.3:** Pros and Cons of Data Access Technologies

<b>Database Access Technologies</b>	<b>Advantages</b>	<b>Disadvantages</b>	<b>Others</b>
Extensible Markup Language (XML)	<ul style="list-style-type: none"> <li>• It allows writing of our own markup language.</li> <li>• We are free to develop tools that meet our needs exactly.</li> <li>• By defining our own tags.</li> <li>• Not be limited to formatting rules.</li> <li>• Interoperability for programs.</li> <li>• Standard.</li> <li>• XML is easy to convert further into different formats as required.</li> </ul>	<ul style="list-style-type: none"> <li>• One of the biggest drawbacks of XML is that it is lacking in the area of adequate applications for processing.</li> <li>• There are no XML browsers on the market yet.</li> </ul>	language

	(see, 4.9.4)	<ul style="list-style-type: none"> <li>• Thus, XML documents must either be converted into HTML before distribution or converting it to HTML on-the-fly by middleware</li> </ul>	
JDBC Type4 Driver	<ul style="list-style-type: none"> <li>• Leverage Existing Enterprise Data</li> <li>• Leverage Existing Enterprise Data</li> <li>• Zero Configuration for Network Computers</li> <li>• Key Features Full Access to Metadata</li> <li>• Database Connection Identified by URL</li> <li>• Allows a direct call from the client to the database, without the need of client pre configuration</li> </ul>	<ul style="list-style-type: none"> <li>• At client side, a separate driver is needed for each database</li> <li>• Requires the use of protocols proprietary to the DBMS.</li> <li>• Need to load a different driver for each DBMS that it needs to access.</li> </ul>	Application programming interfaces (APIs).
Open Database Connectivity (ODBC)	<ul style="list-style-type: none"> <li>• Multi row fetches by a single function call</li> <li>• Code-page translation libraries</li> <li>• Reporting of a driver's ANSI-conformance level and SQL support</li> <li>• On-demand automatic</li> </ul>	<ul style="list-style-type: none"> <li>• A separate module or driver for each database to be accessed. Library that is dynamically connected to the</li> </ul>	Application programming interfaces (APIs).

	<p>population of implementation parameter descriptor</p> <ul style="list-style-type: none"> <li>• Enhanced diagnostics and row and parameter status arrays</li> <li>• Date time, interval, numeric/decimal, and 64-bit integer application buffer types</li> <li>• Asynchronous execution</li> <li>• Stored procedure support, including escape sequences, output parameter binding mechanisms, and catalogue functions</li> <li>• Connection enhancements including support for connection attributes and attribute browsing</li> </ul>	<p>application.</p> <ul style="list-style-type: none"> <li>• ODBC is slower due to all of its components</li> <li>• ODBC still employs some limitations. It can only be used for relational databases and as industry has grown towards object-oriented programming and design, ODBC is being increasingly neglected for its lack of OO support (even though it supports JAVA objects).</li> <li>• Some people complain that Microsoft uses ODBC for its own personal gain.</li> </ul>	
ADO.NET	<ul style="list-style-type: none"> <li>• Performance – there is no doubt that ADO.NET is extremely fast.</li> </ul>	<ul style="list-style-type: none"> <li>• Managed-Only Access, so should be</li> </ul>	Works under .NET environment

	<ul style="list-style-type: none"> <li>• Optimized SQL Provider.</li> <li>• XML Support (and Reliance).</li> <li>• Disconnected Operation Model</li> </ul>	<p>running under the CLR.</p> <ul style="list-style-type: none"> <li>• Only Three Managed Data Providers (so far) – unfortunately, if you need to access any data that requires a driver that cannot be used through either an OLEDB provider or the SQL Server Data Provider, then you may be out of luck.</li> </ul>	
Microsoft.NET	<ul style="list-style-type: none"> <li>• Generation of highly distributed applications.</li> <li>• Simplified software development.</li> <li>• Better user interface over the Web.</li> <li>• Simplified Deployment.</li> <li>• Support for a variety of languages.</li> <li>• An extendable platform for the future</li> <li>• Future portability of compiled applications.</li> <li>• Simple, Funny and Easy to</li> </ul>	<ul style="list-style-type: none"> <li>• Applications running in a managed environment tend to require more system resources than similar applications that access machine resources more directly.</li> <li>• There is concern over</li> </ul>	Development and execution environment that allows different programming languages and libraries to work together



	<p>Learn</p> <ul style="list-style-type: none"> <li>• Enhanced programming model.</li> <li>• Enhanced Safety and Security Managed code.</li> <li>• Common Development Environment Database development.</li> <li>• Better Performance Some functions.</li> <li>• Language Richness Visual Basic, Visual C#, and Visual C++.</li> <li>• Reusability of Code A library.</li> <li>• Extensibility Using Visual Basic, Visual C#, or Visual C++.</li> <li>• Richer developer experience.</li> <li>• Stability and reliability.</li> <li>• Stored Procedures and Triggers &amp; Functions.</li> <li>• Faster development (less to do, the system handlers more).</li> <li>• More reuse because of inheritance.</li> <li>• Higher scalability – many capabilities to help applications scale are built in .NET.</li> <li>• Easier to build sophisticated development tools –</li> </ul>	<p>possible loss of trade secrets and the bypassing of license control mechanisms.</p> <ul style="list-style-type: none"> <li>• Sometimes the frame cannot install in older Windows versions.</li> </ul>	
--	---	--	--

	<p>debuggers and profilers can target the Common Language Runtime, and thus became accessible to all .NET –enabled language.</p> <ul style="list-style-type: none"> <li>• Fewer bugs – whole classes of bugs should be unknown in .NET. with the CLR handling memory management, memory leaks should be a thing of the past, for example.</li> <li>• Potentially better performance – Microsoft’s heavy investment in system level code for memory management, garbage collection, and the like have yielded an architecture that should meet or exceed performance of typical COM-based applications today</li> </ul>		
--	--	--	--

By way of comparison, results from this project as shown in table 4.3 supports similar effort by Suresh (2009). Similarly, table 4.4 shows match J2EE/JDBC .NET /VB.NET on nearly equal ranks. However, this contrast depends on criteria’s such as ease of use, scalability, single language multiple platforms, multiple languages single platform, reliability, performance, speed of development, reuse and open standard.

**Table 4.4: J2EE/JDBC vs .NET/VB.NET**

<b>Criteria</b>	<b>J2EE/JDBC</b>	<b>.NET /VB.NET</b>	<b>Comments</b>
Ease Of Use (Development Environment)	**	****	VB.net and C# are easier to use than J2EE
Scalability	***	**	Execute Java Code on Mainframe
Single Language Multiple Platforms	****	*	Java Can run on many platforms through the JVM
Multiple Languages Single Platform	*	****	VB,C#,J# all run in the same run-time environment
Reliability	**	****	VB/Com development in 1993
Performance	***	***	Equal Performance
Speed of development	*	***	VB code easier to learn
Reuse	****	**	Deploy same code on multiple platforms and multiple projects
Open Standards	*****	*	Java, JVM are open standards
Overall	25 points	24 points	

#### 4.12 Recommendations

In this project various Database Access Technologies have been reviewed in order to recommend the one most suitable to implement. Each method has particular advantages and disadvantages. Actually, the decision depends on some restrictions and operational environment. These include environment (scope), advantages, disadvantages, client and network capabilities, availability, usability, and system performance to the technologies, etc. System performance could be affected by elements such as CPU speed, data size, operating system, etc.

However, it could be recommended that Database Access Technology is required to satisfy the desired project objectives and goals. For instance, it should

have capabilities to allow data confidentiality from unauthorized exposure. That can be met through encryption of the database column level.

From the summary of information about DATs capability supports (Internet, Network and Encryption/Decryption), comparisons on tables (4.2), (4.3) and (4.4) played an important role on the technology selection decision. It was found that the best support on contrast criteria comes from JDBC and Microsoft.NET, while others show some flaws to support the comparative criteria's.

Moreover, project scope is considered as a significant factor on the recommendation decision. However, the best DATs that have comprehensive features and capabilities may be incompatible to some systems. When trying to choose between whether these features are important for the organization, consider the quality of the developers. If they are well-educated and do not require much hand-holding, then they will likely find the flexibility and performance gains from a JAVA/JDBC system as valuable. If the developers require more hand-holding, then the Microsoft.NET approach is clearly superior. Also, the choice between using a native driver or a managed driver depends on the application/development requirements. If the application does not need to be written in Java, then a native driver is probably better suited for such needs. If the application needs to be written in Java, then one can either use a JDBC to ODBC bridge driver or a JDBC driver completely written in Java. The ultimate choice usually depends not on technical superiority, but on:

- Cultural/political preferences
- Customer preference
- Vendor relations
- Cost
- Platform Dependency
- Skill set of your developers

So what's a company like us to do? Both platforms are useful, and both can lead to the same destination. Which platform is right for us? When deciding, we recommend concentration on the larger business issues. We think about the existing developer skill sets, his existing systems, the existing vendor relationships, and customers. These factors always drive the decision, not the minor features.

The project therefore decides to use '*local, platform-specific interfaces*' offers more flexibility than browser-based interface. Microsoft.NET aims to make those two types of interfaces as similar in development philosophy as possible Barwell, *et al.* (2001c). It is recommended to use DOT.NET/VB in this project. However, Microsoft has strong tools vendor at all times. As part of its launch of .NET, Microsoft released a beta version of the Visual Studio.NET integrated development environment. Visual Studio.NET supports all languages supported by earlier releases of Visual Studio - with the notable exception of Java. In its place, the IDE supports C# and VB.net, Microsoft's new object-oriented programming language, which bears a remarkable resemblance to Java. Visual Studio.NET has some interesting productivity features including Web Forms, a web-based version of Win Forms, .NET's GUI component set. Visual Studio.NET enables developers to take advantage of .NET's support for cross-language inheritance.

#### **4.13 Summary**

This Chapter introduced the Database Access Technologies (DATs), and what could be the best to use. Then reviewed analyzed and evaluated DATs definition, features/capabilities, how they work, architecture, advantages and disadvantages. Subsequently, a comprehensive comparison was carried on some aspects of DATs such as internet, network database column level encryption/decryption support, ease of use, scalability, single language multiple platforms, multiple languages single platform, reliability, performance, speed of development, reuse and open standard. Finally, a discussion concludes this chapter with a final decision on the choice of appropriate DATs.

## CHAPTER 5

### SYSTEM DESIGN

#### 5.1 Introduction

This chapter introduces the system design for the reason of describing system component and their relations with each other. It covers in-depth explanations of the components and internal jobs process.

The following sections are organized as follows: section 5.2 covers the user requirements, section 5.3 deals with database design, section 5.4 concerns system requirements and the last section 5.5 is summary.

#### 5.2 User Requirements

During the initial stages of the project development, a list of requirements was identified and they are arranged according to the module they belong to, as below:

- Initialization.
- Login.
- Main Form
- Manage User Information.
- Encryption

- Decryption

The following is a brief of discussion on the requirements that relate to the project scope.

### **5.2.1 Initialization**

Secure Column Application initialization occurs only one time (the first run of the system). The Secure Column Application should provide an interactive and easy to use interface for the user to carry out the initialization process. The user must feed valid login information to connect to the SQL Server 2005, and then the Secure Column shall load the available databases to the server. After selection of the default database is made by the user, the Secure Column sets up the necessary system information into the selected database schema (Scolumn) and the tables used by the system. To complete the initialization process the Secure Column Application prompts the user to feed his/her password to generate the administrator's key.

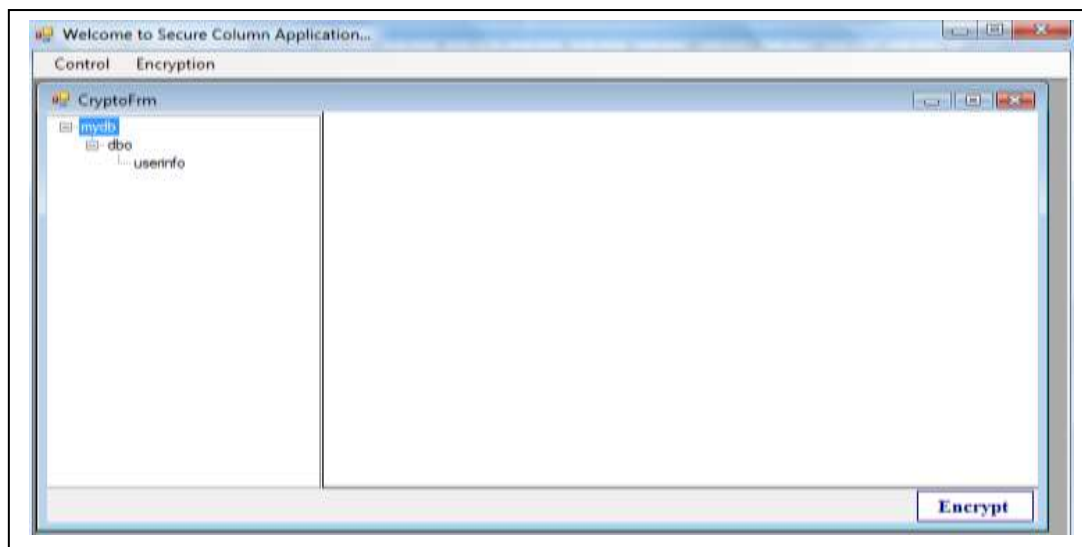
### **5.2.2 Login**

The Secure Column Application should provide the user the appropriate interfaces to either login to the Secure Column Application or change his/her password. In either case the user must feed valid information, otherwise the Secure Column Application will not allow the user to login into the system. There are two types of login:-

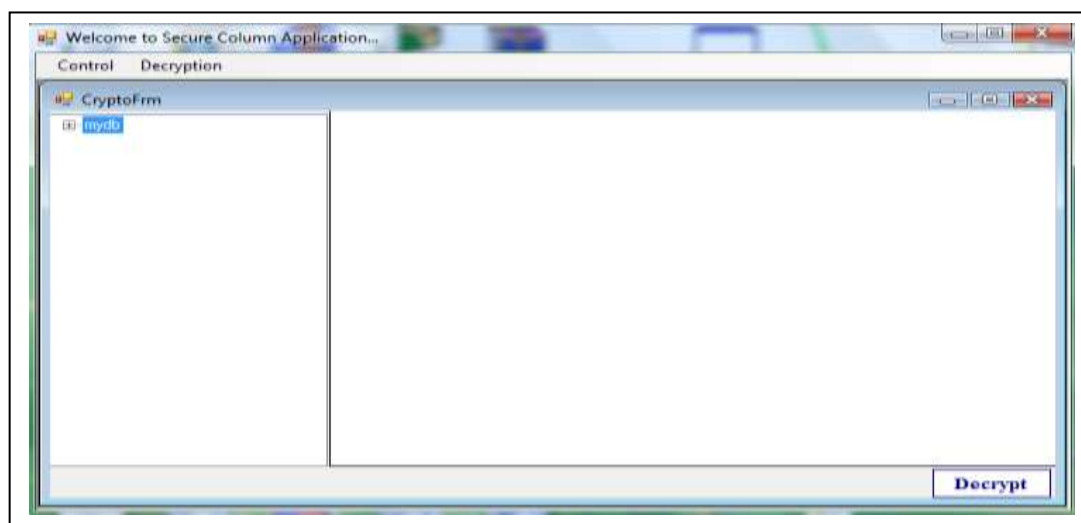
- To the SQL Server 2005 Express Edition.
- To the Secure Column Application.

### 5.2.3 Main Form

The main forms are classified in two systems, Encryption Main Form (EMF) and Decryption Main Form (DMF). The EMF is used by Security Manager to perform encryption as shown in figure (5.1). The DMF is used by Data Owner to do decryption as shown in figure (5.2).



**Figure (5.1):** Encryption Main Form



**Figure (5.2):** Decryption Main Form



#### **5.2.4 Manage User Information**

The Secure Column Application should provide a user friendly interface that enables the administrator to add database user to the Secure Column Application, manage user information (grant database access to user, change the system status of the user), delete user from the Secure Column Application.

#### **5.2.5 Encryption**

During encryption the following requirements should be met:

- A user friendly GUI window for encryption purpose shows the logged in user's database(s)/schemas(s) and table(s) that are accessible by he/she.
- The AES Rijndael algorithm should be used as symmetric cryptography for data encryption.
- A proper authentication and validation of the user is most.
- Verification of the encryption process, acknowledgement and log the user activities are required.

#### **5.2.6 Decryption**

During decryption the following requirements should be done:

- A user friendly GUI window for encryption purpose shows the logged in user's database(s)/schemas(s) and table(s), which are accessible by he/she.
- The AES Rijndael algorithm should be used as symmetric cryptography for data encryption.
- The transparent mechanism should be updated as required.
- A proper authentication and validation of the user is most.

- Verification of the decryption process, acknowledgement and log the user activities are required.

### 5.3 Database Design

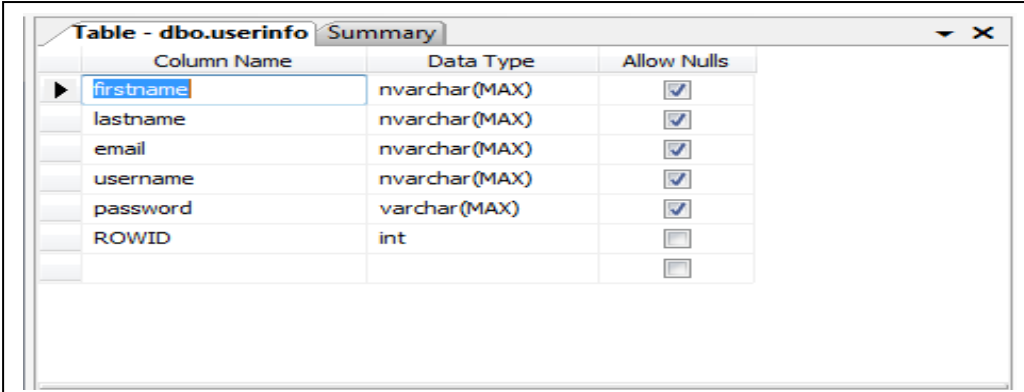
Database design produces a detailed data model of a database. Secure Column Application targets Microsoft SQL Server 2005 as a development and deployment environment. The Secure Column Application secures the above mentioned relational database management system RDBMS from unauthorized access and makes sure of the integrity, confidentiality and availability of the data to the authorized users only.

Secure Column Application's data storage is designed in a schema named Scolumn which is accessible only to the users. In the following subchapters, detail information on Secure Column Application's database design such as the tables used by the schema will be provided.

#### 5.3.1 DBO.USERINFO

The DBO.USERINFO table contains the details of the user information which is necessary for the login. The table below (5.1) shows the design of the table.

**Table 5.1:** DBO.USERINFO Table Design

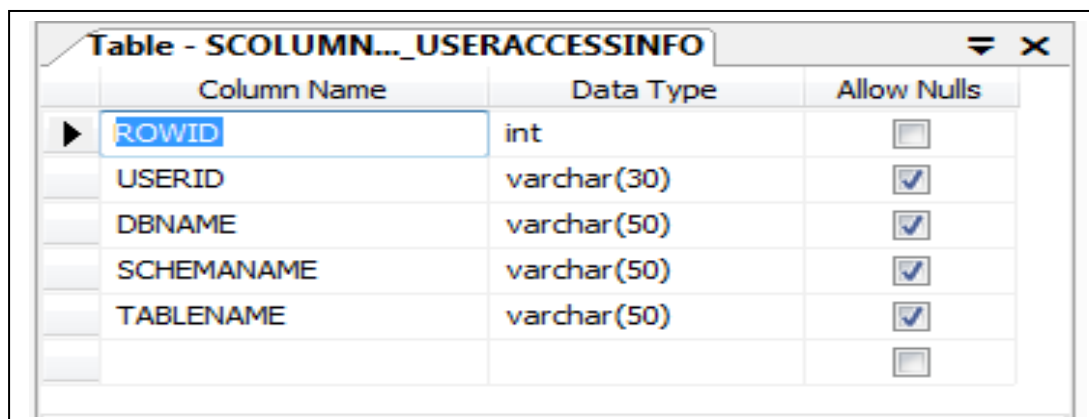


Column Name	Data Type	Allow Nulls
firstname	nvarchar(MAX)	<input checked="" type="checkbox"/>
lastname	nvarchar(MAX)	<input checked="" type="checkbox"/>
email	nvarchar(MAX)	<input checked="" type="checkbox"/>
username	nvarchar(MAX)	<input checked="" type="checkbox"/>
password	varchar(MAX)	<input checked="" type="checkbox"/>
ROWID	int	<input type="checkbox"/>

### 5.3.2 SCOLUMN.SEC\_USERACCESSINFO

The SCOLUMN.SEC\_USERACCESSINFO table contains the list of database objects a user is authorized to access them. The table below (5.2) shows the design of the table.

**Table 5.2:** SCOLUMN.SEC\_USERACCESSINFO TABLE DESIGN

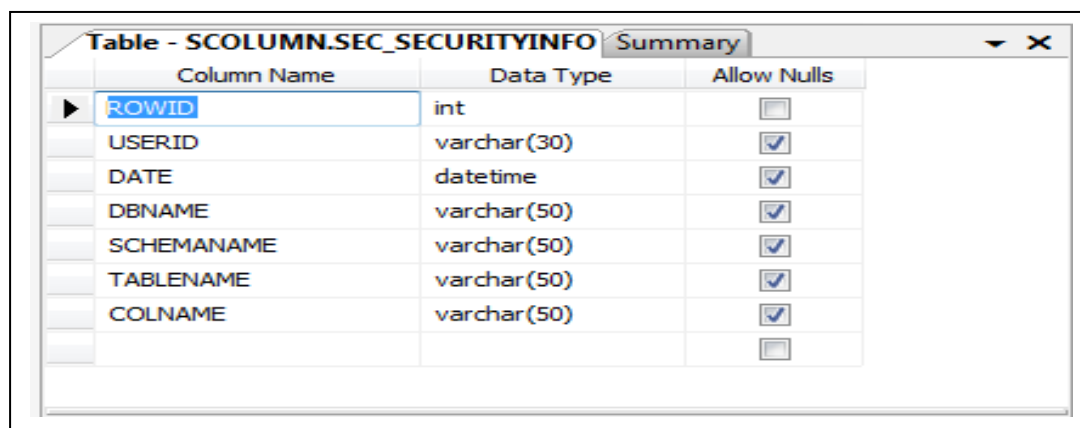


Column Name	Data Type	Allow Nulls
ROWID	int	<input type="checkbox"/>
USERID	varchar(30)	<input checked="" type="checkbox"/>
DBNAME	varchar(50)	<input checked="" type="checkbox"/>
SCHEMANAME	varchar(50)	<input checked="" type="checkbox"/>
TABLENAME	varchar(50)	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

### 5.3.3 SCOLUMN.SEC\_SECURITYINFO

The SCOLUMN.SEC\_SECURITYINFO table contains information about the column encrypted or decrypted through the Secure Column System. The table (5.3) shows the design of the table.

**Table 5.3:** SCOLUMN.SEC\_SECURITYINFO TABLE DESIGN



Column Name	Data Type	Allow Nulls
ROWID	int	<input type="checkbox"/>
USERID	varchar(30)	<input checked="" type="checkbox"/>
DATE	datetime	<input checked="" type="checkbox"/>
DBNAME	varchar(50)	<input checked="" type="checkbox"/>
SCHEMANAME	varchar(50)	<input checked="" type="checkbox"/>
TABLENAME	varchar(50)	<input checked="" type="checkbox"/>
COLNAME	varchar(50)	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

### 5.3.4 SCOLUMN.SEC\_CRYPTINFO

The SCOLUMN.SEC\_CRYPTINFO table stores the details of the encrypted data. The table (5.4) shows the design of the table.

**Table 5.4: SCOLUMN.SEC\_CRYPTINFO TABLE DESIGN**

Column Name	Data Type	Allow Nulls
ROWID	int	<input type="checkbox"/>
USERID	varchar(50)	<input checked="" type="checkbox"/>
DBNAME	varchar(50)	<input checked="" type="checkbox"/>
SCHEMANAME	varchar(50)	<input checked="" type="checkbox"/>
TABLENAME	varchar(50)	<input checked="" type="checkbox"/>
COLNAME	varchar(50)	<input checked="" type="checkbox"/>
ROWNO	varchar(100)	<input checked="" type="checkbox"/>
SMSK	varchar(MAX)	<input checked="" type="checkbox"/>
ADMINISK	varchar(MAX)	<input checked="" type="checkbox"/>
AUSK	varchar(MAX)	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

## 5.4 System Requirements

To use the fully functional of Secure Column Application, the following are prerequisites as related to the hardware and software needs of the system:

### 5.4.1 Hardware Requirement Specification

The hardware requirements are vital for the system to function. Here project requires the following minimum hardware specification:

The specification can be summarized in the following table (Table 5.5):

**Table 5.5:** The Hardware Specifications

<b>Component</b>	<b>Requirement</b>
Computer processor	Intel Pentium 1024GHz or above.
Memory	1 GB or more.
Hard disk	1.2 GB for installations; 2.9 for CD install
Drive	CD-ROM or DVD-ROM drive
Display	VGA or hardware that supports console redirection required; Super VGA supporting 800 x 600 or higher resolution monitor recommended

#### 5.4.2 Software Requirement Specification

The software requirements here mean the operating environment required by the Secure Column Application to fully operate. Following are the software requirements of the Secure Column Application.

- Operating System: Microsoft Windows XP
- Microsoft Visual studio.Net
- Microsoft SQL Server 2005 Express Edition

#### 5.5 Summary

This chapter covered the technical details of the system design. Each component of the overall system was described in some details. The most important component of the system is the User Requirements (Initialization, Login, Main Form, Manage User Information, Encryption and Decryption), Database Design and system requirements. In the next chapter, the practical implementation details are covered.

## CHAPTER 6

### SYSTEM IMPLEMENTATION AND TESTING

#### 6.1 Introduction

This chapter is intended to cover the on the whole system implementation and testing. A description of the system design was discussed in Chapter 5. Therefore, in this chapter, a detailed implementation of the system design is conducted. Chapter five outlines each component of the previous design and its implementation.

#### 6.2 Coding Steps:

In this section, the major modules that belong to the scope of my work in the Secure Column Application will discussed:

##### 6.2.1 Initialization

Initialization part consists of the initialization and authentication processes of the Secure Column Application. Below are the classes involved in those processes:

- SysLoginFrm Class: this class is responsible for the initialization of the system for the first time and the system and database authentication.

Through the initialization module, the user should enter his/her database login information to connect to the server and then selects the default database where information resides, after that the user must provide his/her system password to complete the initialization module.

### **6.2.2 User and key management unit:**

This unit comprises of all the characteristics that relate to the user and key management, such as adding new user to the system, granting access privileges to the newly added user, generating keys for the user and removing keys from the system. Following are the classes that involve to this module:

- AddUserFrm Set: this class is used to add users (Security Managers, Application Users) to the Secure Column Application and to grant them access to the desired databases.
- ManageUserFrm Set: this class is used for user and key management: key generation, changing system status and user deletion functionalities are provided by this class.

Secure Column Application users are concerned by any user who will use the system, and characterized as the following:

- Administrator: is the System administrator, responsible to control the system and manage other users. The administrator can add users to the system generate their keys and delete them.
- Security Manager: is responsible to secure the sensitive data in the database;
- Application User: represents the GUI user that accesses the database secured by Secure Column Application.

### 6.2.3 Test and Evaluation the result

After the objectives of the testing are identified, several test cases to be performed in order to realize the software requirements and to make sure that the system is doing what it is supposed to do. However, the below tables (6.1) and (6.2) will show the test approaches and test results sequentially:

**Table 6.1:** Test Approaches of Secure Column Application.

No	Features	TestNo
1	System Initialization	INI
2	User and Key Info Management	UKIM
3	Login	LOG

**Table 6.2:** Test Result of Secure Column Application Test

TestNo	Features	Description	Result
INI	System Initialization		
	1	Establish Connection to Database	PASS
	2	System Administrator's Keys and Login account generation	PASS
UKIM	User and Key Info Management		
	1	Create Secure Column User	PASS
	2	Display User Information	PASS
	3	Activate/Deactivate the User	PASS
	4	Delete User from the Secure Column Application	PASS
LOG	System Login		
	1	Login to Database System	PASS
	2	Login to Secure Column Application	PASS



## 6.3 Administrator and User Manual

This part provides a brief steps for the Secure Column System Administrator, by offering some screen shots with short description.

### 6.3.1 Create Users by System Administrator

This section explains how to the System Administrator of Secure Column Application can create System Security and Data Owner users, and grant them some privileges. These steps are as follows.

#### 6.3.1.1 Run the SQL Server 2005 Express Edition

As shown in figure (6.1) bellow, the steps are as follows:

- Select Start
- Program Manager
- Microsoft SQL Server 2005
- SQL Server Management Studio Express



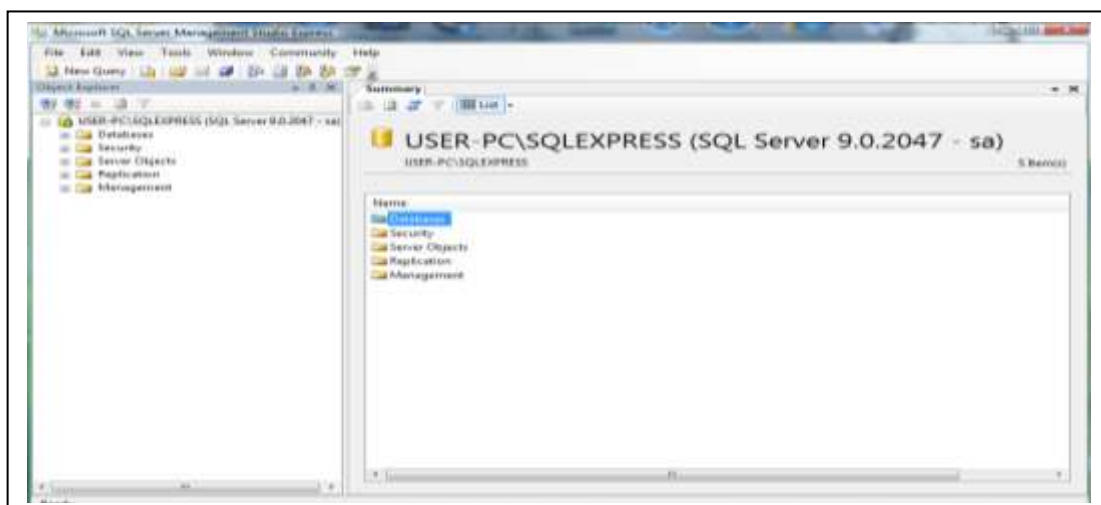
**Figure (6.1):** Run SQL Server 2005 Express Edition.

The following figure (6.2) shows how to complete Server Connection; System Administrator authentication is required, for instance Login: and Password should enter correctly.



**Figure (6.2):** Server connection by System Administrator

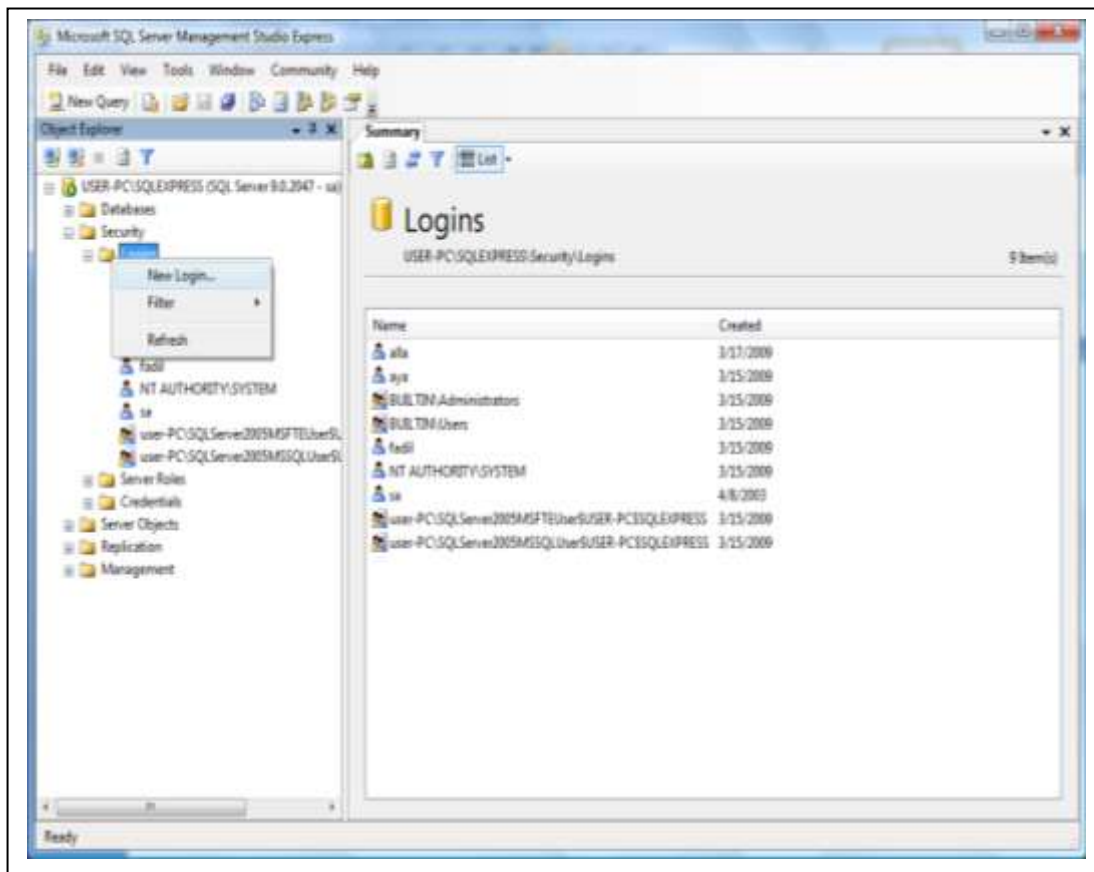
The following figure (6.3) shows the management options; however, the System Administrator will create users and grant some privileges to them. The essential classes for the System Manager (sa) are Databases and Security.



**Figure (6.3):** Management Options

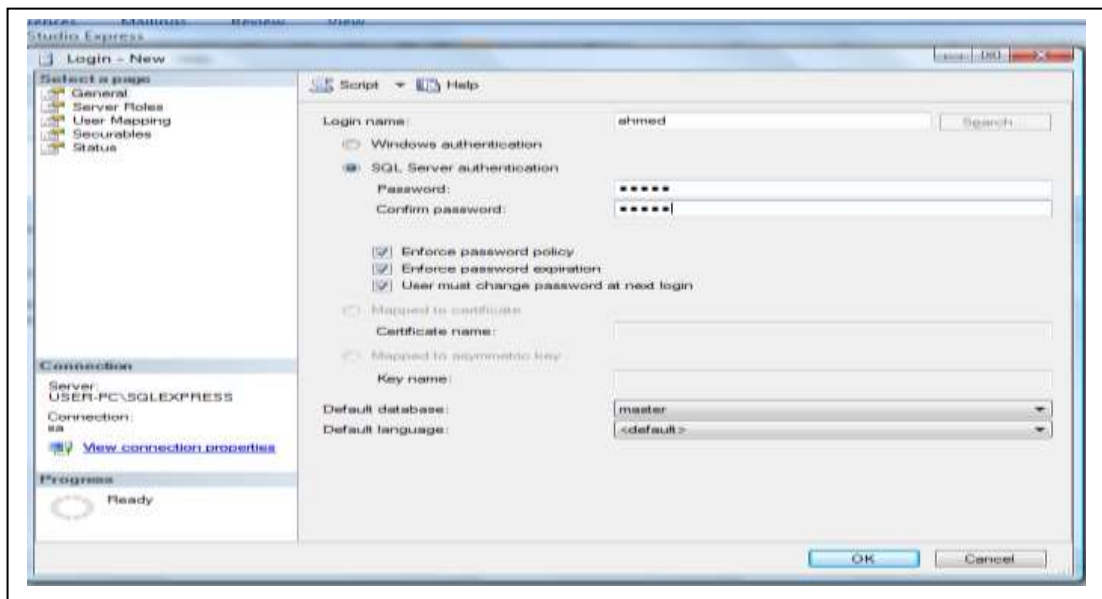
From Security class, the System Manager can create users through following these steps as shown in figure (6.4) bellow.

1. Security
2. Login
3. Press the right click mouse on Login and select New Login



**Figure (6.4):** Create New Login

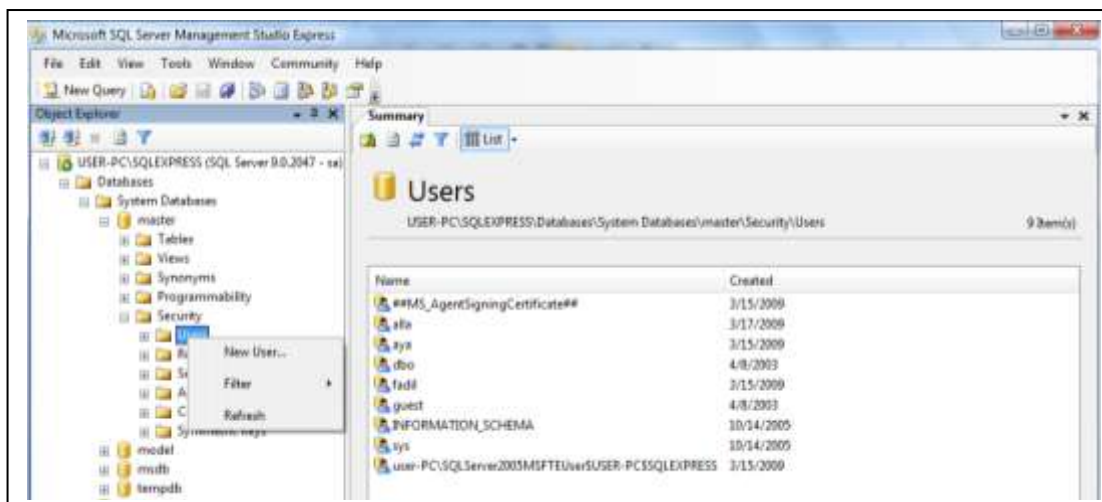
4. Add the user name in Login name space.
5. Select SQL Server authentication and add Password to user, as shown in figure (6.5) bellow.



**Figure (6.5):** Login-New

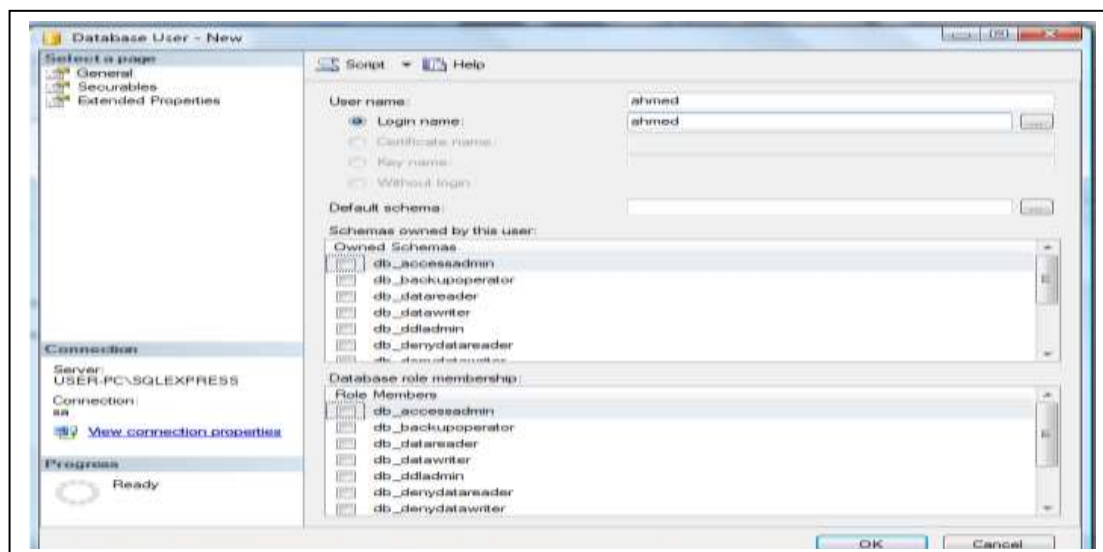
Nevertheless, from Databases we can define the New user to the database by the following steps:

1. Databases\System Databases\Security
2. Press right click mouse on security and select New User, as shown in the following figure (6.6) bellow.



**Figure (6.6):** Create New user on Database.

3. Add the User name and Login name
4. Ok, as shown in the figure (6.7) bellow.

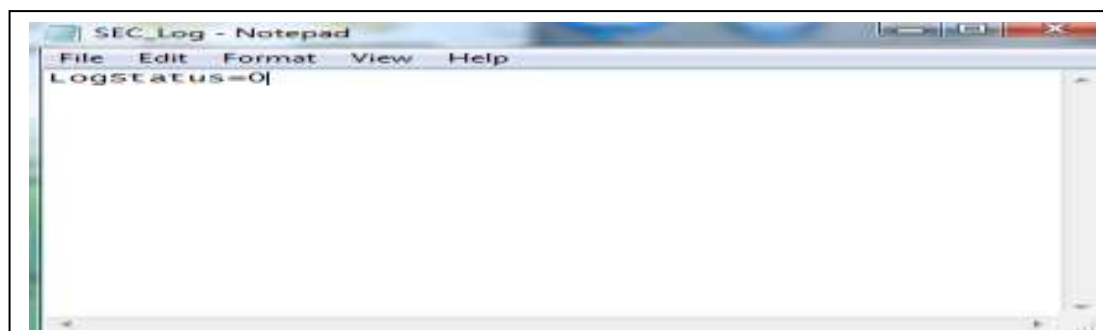


**Figure (6.7):** Add user name and Login name.

### 6.3.1.2 How to make Initialization

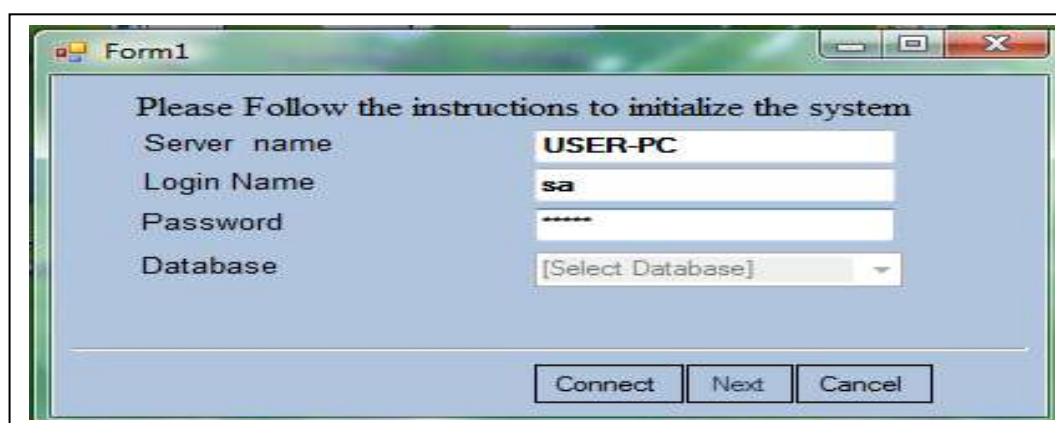
System initialization: when the user runs the system for the first time, the initialization form appears, to use encryption/decryption system the user must initialize Secure Column Application as following the instructions on figures.

- Firstly, change LogStatus to equal 0, to make resetting, as shown in figure (6.8) bellow.

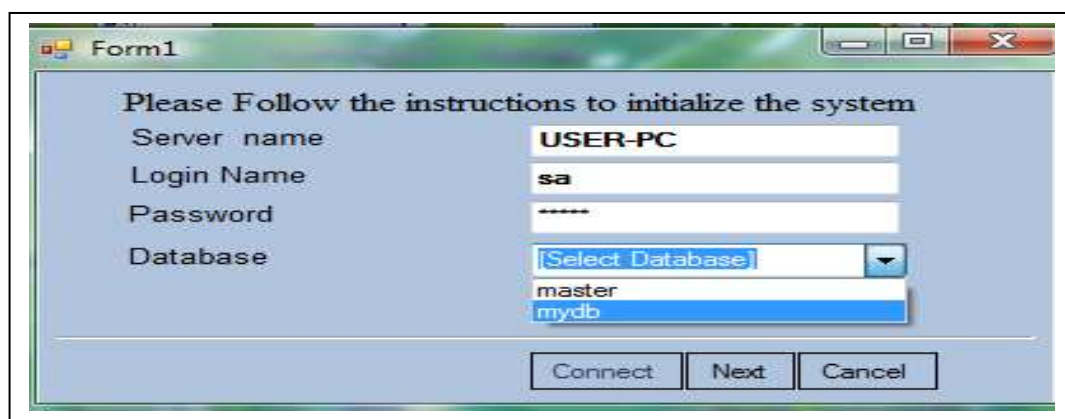


**Figure (6.8):** Log Status File

- The following figures (6.9) and (6.10) respectively are used by Database Administrator to complete the login and Initialization. Moreover, the Database is recognizes automatically.
  1. Server name = USER-PC
  2. Login Name = sa
  3. Password= \*\*\*\*\*
  4. Then Connect
  5. Then select Database (mydb)
  6. Next



**Figure (6.9):** System Administrator Authentication



**Figure (6.10):** Automatic database detection

Then the bellow figure (6.11) is used to confirm Admin's Login Account as follows:

1. User ID = sa

2. Password = \*\*\*\*\*
3. Re-enter Password = \*\*\*\*\*
4. Done



The screenshot shows a window titled 'MKeyfrm' with a light blue background. At the top, it says 'Fill the below fields to create Admin's Login Account.' Below this, there are three input fields: 'User ID' with the text 'sa', 'Password' with six asterisks, and 'Re-Enter Password' with six asterisks. To the right of the password fields, it says 'Min 6 Characters'. At the bottom, there are two buttons: 'Done' and 'Cancel'.

**Figure (2.11):** Admin's Login Account

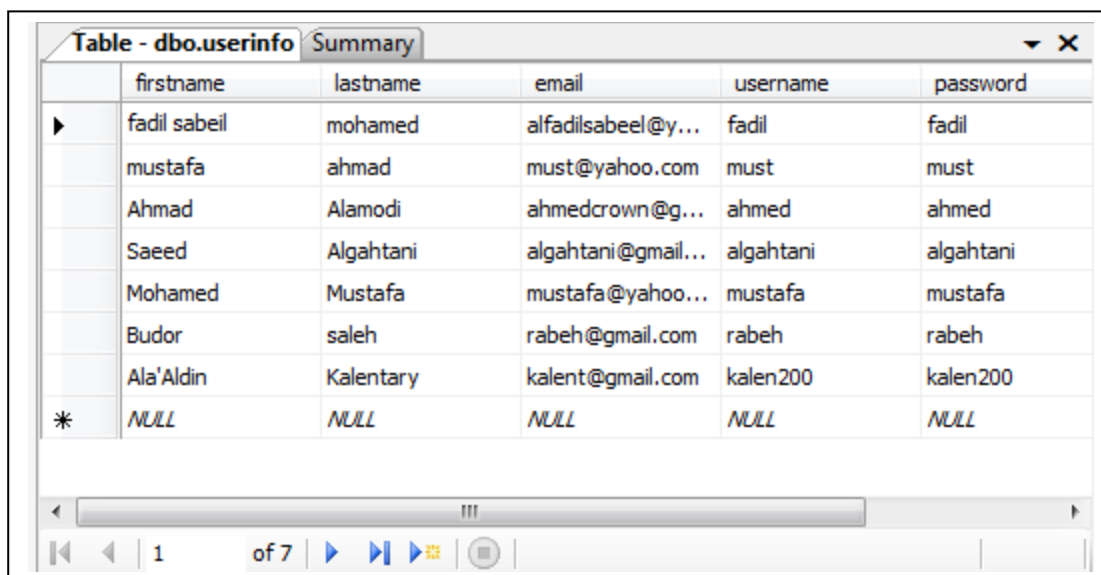
When you have the following figure (6.12), the initialization is successful, and this is the initialization end.



**Figure (6.12):** Initialization successful

### 6.3.1.3 How to make Encryption

The following figure (6.13) shows the dbo.userinfo table information before encryption some columns.



	firstname	lastname	email	username	password
▶	fadil sabeil	mohamed	alfadilsabeel@y...	fadil	fadil
	mustafa	ahmad	must@yahoo.com	must	must
	Ahmad	Alamodi	ahmedcrown@g...	ahmed	ahmed
	Saeed	Algahtani	algahtani@gmail...	algahtani	algahtani
	Mohamed	Mustafa	mustafa@yahoo...	mustafa	mustafa
	Budor	saleh	rabeh@gmail.com	rabeh	rabeh
	Ala'Aldin	Kalentry	kalent@gmail.com	kalen200	kalen200
*	NULL	NULL	NULL	NULL	NULL

**Figure (6.13):** Dbo.userinfo table before encryption.

To make encryption to previous table, only the Security Manager (fadil) can apply these steps:

- The first steps show Database Login Information authentication figure (6.14):
  1. Login ID = fadil
  2. Password = \*\*\*\*\*



**figure (6.14):** Database Login authentication

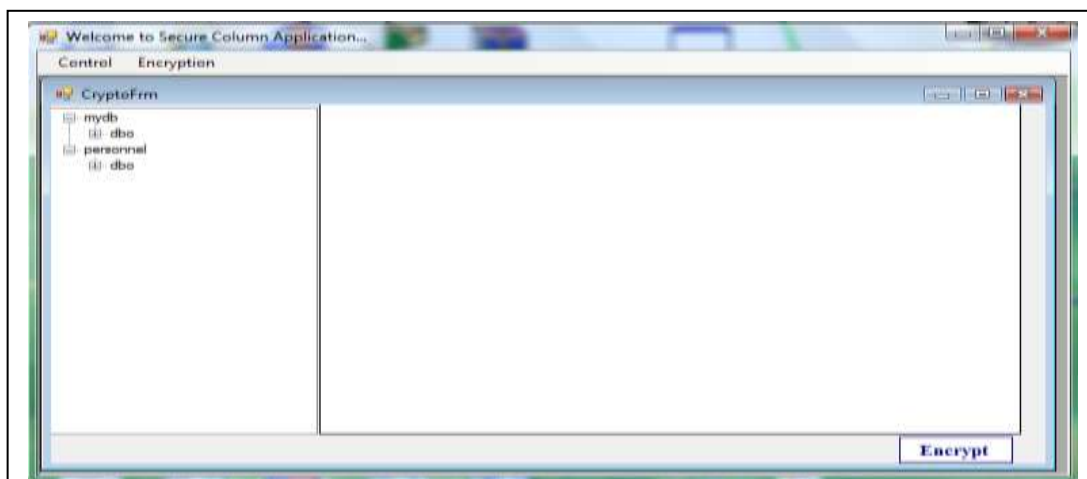


- Then the next is System Login Information Authentication as follows in figure (6.15):
  1. User ID = fadil
  2. Password = \*\*\*\*\*
  3. Login

A screenshot of a Windows-style application window titled "Form1". The window has a light blue background and a title bar with standard minimize, maximize, and close buttons. The main content area contains the text "Please Enter valid System Login Information" centered at the top. Below this, there are two input fields: "User ID" with the text "fadil" entered, and "Password" with a masked password of "\*\*\*\*\*". At the bottom of the form, there are three buttons: "Login", "Change Password", and "Cancel".

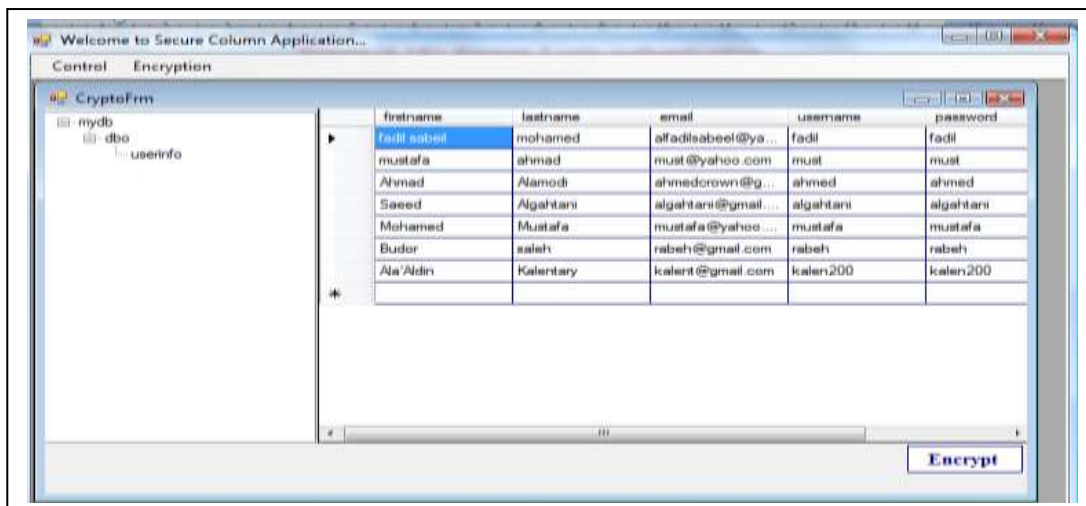
**Figure (6.15):** Encryption System Login authentication

However, the following platform figure (6.16) appears, and automatically displaying all databases on the system. The security manager can do encryption to any database.

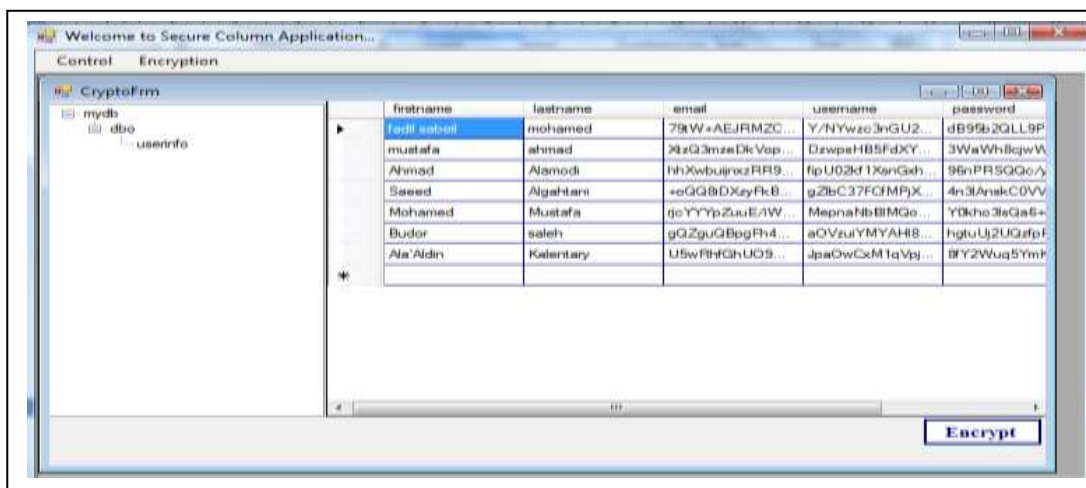


**Figure (6.16):** Display of all databases automatically

Then the following figure (6.17) appears; to achieve encryption to any column, select the required column and press Encrypt Button. The result is shown in figure (6.17).



**Figure (6.17):** The data before encryption.



**Figure (6.18):** Columns Encryption.

Figure (6.18) shows encryption of some columns such as email, username and password columns. Nevertheless, all these encryption was done by the Security Manager (fadi).

### 6.3.1.4 How to make Decryption

Encryption is a process opposite to encryption. However, the columns are encrypted by Security Manager, only the Data Owner can do decryption. Then following steps are required.

- Data Owner Login for both Database and System are shown in figures (6.19) and (6.19) respectively.

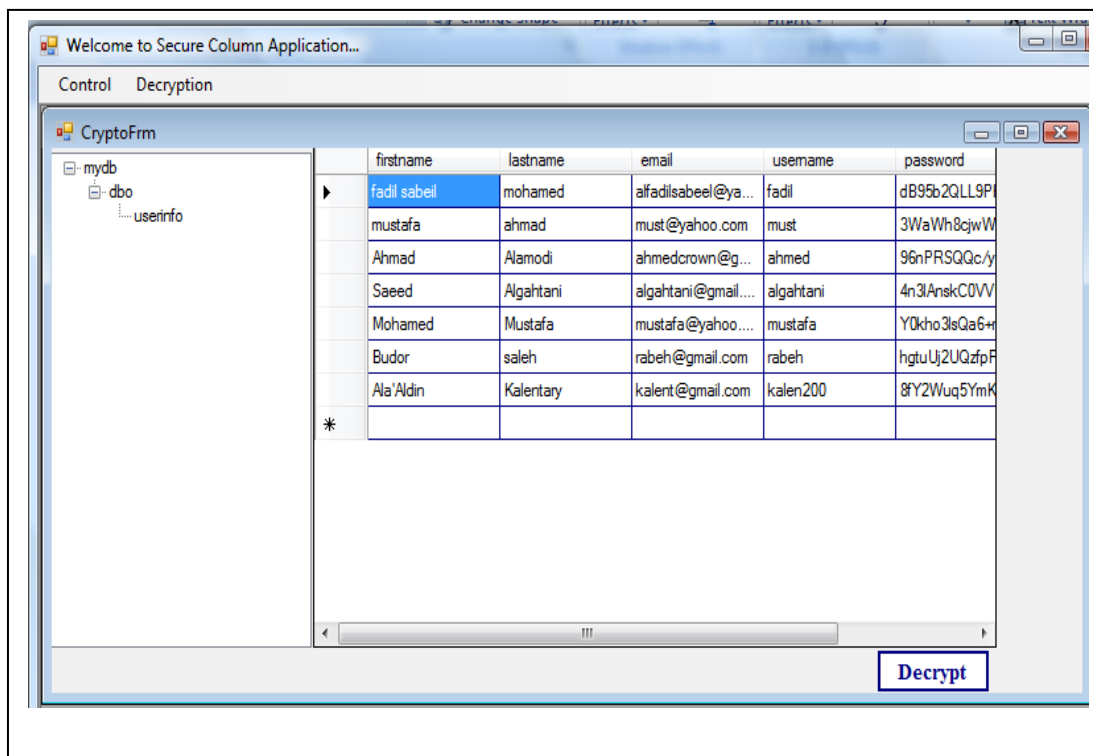


**Figure (6.19):** Data user Login to the Database



**Figure (6.20):** Data user Login to the Encryption System.

Then the following figure (6.20) appears to offer decryption facilities to the Data Owner. To make decryption, select the required column and press the Decryption Button. For instance, one column is decrypted as shown in figure (6.21).



**Figure (6.21):** Columns decryption Result.

#### 6.4 Summary

This chapter contains the technical implementation of the system. The explanation was enriched with the step-by-step implementation. Secure Column Application functions were described. Finally, the overall system was tested with registration of a real user account.

## CHAPTER 7

### BENEFITS, DISCUSSION AND FUTURE WORKS

#### 7.1 Introduction

This last chapter of the research mainly discusses the implemented system, which could lead to future works necessary to enhance the effective implementation. The chapter is in seven sections. Following this introduction is section 7.2 where an implementation modules. Section 7.3 is lessons learnt. Section 7.4 is an Expected Organizational Benefits. Section 7.5 deals with suggestions for the further research in the area of level security schemes for database including cryptography and database security in common. Finally, the chapter concludes with section 7.6.

#### 7.2 Implemented Modules

This project discussed the issue of database column level encryption through DATs, study analysis, comparison and recommendation of suitable DAT to be used; in order to meet the project objectives, scope and purposes. Throughout the course of conducting this research, we have made several progress and achievements particularly in the design and implementations of a database column level security scheme based on encryption technology, Database Access Technology, and address related and relevant issues. In the following paragraphs, presents views on the advancements made so far in the research and its contributions towards the body of knowledge. The discussion in this Section focuses on five points as follows:

- Improvement of database column level encryption - decryption implementation. The first objective of the research is to design and implement a security scheme of DBMs. This objective is achieved with the design of a new column level security scheme for the encryption and decryption of data in databases. The column level security scheme has employed some concepts and tools in the fields of cryptography, data security and using encryption algorithm design utilities of SQL Server. Specially, the research has utilized symmetric key and Initialization Vectors for key generation and for encryption/decryption process. The detailed description and utilization of these concepts have been covered in Chapter 5. In meeting this objective, the research has produced a new database column level security scheme (Chapter 5) supported by key generation, management of the users and login. The new security scheme is able to encrypt/decrypt data at column level of relational database model through using DATs.
- A new approach in accessing and processing of encrypted data. In line with the first objective of the research in improving the using of database column level security, we were able to develop a new approach in accessing and processing of data encryption/decryption. As mentioned in the previous paragraph, this approach is based on the use of data Initialization Vectors (Data\_IVs). Unlike data labeling which has been a popular technique used in deriving multilevel secure databases, we used the data themselves (partial) as their IV that allows controlled access and processing of encrypted data ( zailani, 2004).
- Initialization module: this module encompasses the Secure Column Application initialization feature which is done at the first run of the application.
- Authentication module: this module is implemented through the Database and the Secure Column Application logins to grant the users access to the system;
- User and key management module: this module consists of the implementation of user management features such as adding users to the Secure Column Application, granting the users access control in the

system and removing users from the system, besides the secure cryptographic key generation for the users.

### **7.3 Lessons Learnt**

**Database Security Principles:** In this digital era, information has a significant impact on different aspects of our life, and the protection of the information from malicious or unauthorized access is equally essential. Thus, we need to secure the information from unlawful people (hackers). Consequently, the use of cryptography is crucial for protecting the data or information. In this project the author gained the basic and fundamentals of the cryptography, its characteristics and pillars. Most importantly, the author gained an experience of how to implement the cryptography in the column level database systems (SQL Server 2005) in this project.

### **7.4 Expected Organizational Benefits**

An important business data in databases is a clear target for attackers. Even though access control has been deployed as a security mechanism for a long time, almost since the birth of large database systems, the security of a database was considered an additional problem to be addressed. This arises due to threats to the confidentiality and veracity of data.

In order to manage internal threats, vendors or contractors and database administrators with free access to data are increasingly viewed as potential threats to highly sensitive electronic data. Encryption/Decryption enforces levels of protection, and separation of duties for preventing the abuse of the privileges granted to the authorized accounts.

Securing of data at on store is achieved through the encryption of information stored in the database since most attacks occur against the end points of data, where

data sits for long periods of time. Without taking care of such threat, the valuable business information will be compromised and that may lead to series problem for both the organization and those who are responsible for safeguarding the information.

Most of the attacks on the databases come internally, from the organization's employees, who have extra knowledge about the database structure and security policy of their organization. If database security solution is applied on the sensitive information, such tragedy could be avoided.

One of the main benefits for the organization in secure the sensitive information is the credibility of the organization as observed by its customers, clients and counterparts. If the organization realizes and manages properly secured classified information, then organization gains liability and wins the trust of its customers. Hence the security of data at rest is crucial and very significant not only to win the heart of the customers but also to avoid the bad consequences and tragedies that may result from unauthorized access to the valuable information of that organization.

Secure Column Application provides a significant solution to this vital problem by providing the organization with easy to use column level database security system, which ensures the safety of the sensitive data at rest. Similarly, the user friendly environment of the system enables the end users (organizations and business infrastructures) to decide what data to secure and who should have access to sensitive data.

Regulations and compliance; with the increasing demands by legislations and regulators for providing better security governance, in protecting the privacy of the data especially customer information, Secure Column Application offers flexibility to comply with these regulations through column database encryption strategy without additional operation costs and minimal end-user intervention. The key organizational benefits are:



- **Grant confidentiality:** Secure Column Application protects the organization data from the reach of unwanted and unauthorized people and makes it available to its owners only.
- **Trim down outside risk:** with the increase of data accessibility from different sources such as internet and intranet, the need of security rises as threads increase. Encryption/Decryption provides a means of allowing increased access to stored information while reducing the risk of attack, unauthorized viewing and staff negligence;
- **Control the “insider threat”:** database administrators with unauthorized access to data are increasingly viewed as potential threats to highly sensitive classified data. Secure Column Application enforces levels of protection, and separation of duties for preventing the abuse of the privileges granted to the authorized accounts.

## **7.5 Future Works**

Secure Column Application is currently requires securing a specific database system which is Microsoft SQL Server 2005. In future, it could be more functional to make the system more comprehensive in the future releases, to other widely used database systems such as DB2, Oracle and MySQL. Also, the system should be implemented on wide area coverage as desired.

## **7.6 Summary**

The Secure Column Application guarantees a greater degree of security of the sensitive data in the database. Its importance stems from the general recognition that database security methods must be applied at the highest level, mostly in databases with very sensitive information. Besides, the Secure Column Application focuses on the key management (generation, distribution and storage) which is the backbone of any security infrastructure.

This project proposes that the initial steps of an approach to securing databases lies in encrypting the sensitive data (any column) in the database. This mechanism appears to provide maximum safety of the critical data by keeping it in encrypted form. Once the sensitive data is encrypted, concerns about disclosure particularly insider attack is minimized.

- Zailani, M. S. (2004). *Design and Implementation of a new multilevel security scheme for database management systems*. Thesis: PhD.Universiti Teknologi Malaysia.
- Microsoft Corporation (2009a). *Microsoft Open Database Connectivity (ODBC)*: [http://msdn.microsoft.com/en-us/library/ms710252\(VS.85\).aspx](http://msdn.microsoft.com/en-us/library/ms710252(VS.85).aspx)
- Wikipedia. (2008). *Open Database Connectivity*  
: <http://en.wikipedia.org/wiki/Odbc>
- Paolo Romano. (2005). *The design and evaluation of novel mechanisms to address Quality of Service (QoS) issues in Web-based transactional applications*. Italy, Universit\_a di Roma \La Sapienza. Doctorate Thesis Report.
- Samuel, S. (2004). *Universal Help Desk Database Access Tool Using C# and WEB Services on the .NET Platform*. USA, Azusa Pacific University:
- B. Frédéricque, S., et al. (2007). *Knowledge-Based Process Management To Populate Databases With 3d Multi-Representation Of Buildings*. Canada, Université Laval, Québec
- Jin-Tsong Hwang. (2008). An Embedded Google Earth/Maps Application On Real Estate Database Inquiry And Display: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*. Vol. XXXVII. Part B4. Beijing 2008
- N. Meyyappan., et al. (2001). Design and Development of a User-Centred Digital Library System. Singapore 639798, Nanyang Technological University
- George, S., et al. (2008). *Cost Efficient Management Tools for Assessing Cultural Resources Project: Arkansas Archeological Survey.US/DOT/NRTSC-03.08*
- Md. Nasir S., et al. (2008). Improved Reinforcement-based Profile Learning for Document Filtering. *IJCSNS International Journal of Computer Science and Network Security*. VOL.8 No.11, November 2008. Malaysia, University Putra Malaysia
- Ikhu-Omoregbe, N. (2008). Designing E-Education Supports In E-Health Based Systems. *Turkish Online Journal Of Distance Education-Tojde* July 2008 Issn 1302-6488 Volume: 9 Number: 3 Article 11
- Ortolf, H., et al. (2005). *Adapt Computers to Archeology or Adapt Archeology to Computers?* :[www.ubi-erat-lupa.org](http://www.ubi-erat-lupa.org)
- Jermu, P., and Juho, R. (2008). *A Neural Network Tool for Brewery Fermentations*. VTT Biotechnology, P.O.Box 1500 (Tietotie 2, Espoo), FIN-02044 VTT, Finland
- Cokhan Aydinli (2000). *Web Based Finance Tools Egarch and the Rex –Client*. Berlin. Humboldt University at Berlin
- Vadim Bichutskiy. (2008). Heterogeneous Biomedical Database Integration Using A Hybrid Strategy. *The Uciundergraduate Research Journal*
- Nick, P., And Hussein, D. (2005). Evaluation of Toll Collection Performance Using Traffic Simulation. *27th Conference Of Australian Institutes Of Transport Research (Cairt 2005)*. The University Of Queensland
- Maribel, S., And Luís, A. (2008). *Knowledge Discovery In Spatial Databases*. Information Systems Department And Algoritmi Research Centre. University Of Minho
- Faisal, I. and Hassan, M. (2003). *Ontology Construction for Structured Textual Data*. Dept. of Electrical & Computer Engineering, University of Illinois at Chicago
- Philippe, Th., et al. (2005). *Database Wrappers Development: Towards Automatic Generation*. Technische Universiteit Eindhoven, The Netherlands. Universit´e de Namur, Belgium.
- Boon, L., and Jake, B. (2003). *System Development – DEVIL Brokerage Utility. Dynamically Enhancing VLE Information from the Library – DEVIL Project*. University of Edinburgh
- Erik Harper. (2006). *Open-Source Technologies In Web-Based Gis And Mapping*. Master of Sc.Thesis. Northwest Missouri State University Maryville, Missouri
- L. Cinque, et al. (2003). *A complete Open Source Architecture for Paper Document Recognition*. Computer Science Department, University”La Sapienza” of Rome Via Salaria 113, 00198 Rome, Italy.

- Fredrik Bökman. (2001). *Data Sources, Exchange Formats, and Database Schemas for a Proteo-Chemometric Analysis and Query System*. Thesis for the Degree of Master of Science Department of Information Science Majoring in Computer Science. Uppsala University
- William J. Hallinan. (2008). *Improving Software Engineering Practice through Competency Based Personnel Reviews*. Master of Science. Thesis. College of Graduate Studies, University of Idaho
- Zhiyuan, Sh., and Hai, J. (2008). *Middleware Based High Performance and High Available Database Cluster*. Huazhong University of Science and Technology, Wuhan, 430074, China
- Daan, L., and Erik, M. (1999). *Domain Specific Embedded Compilers*. Department of Computer Science, University of Utrecht. The Netherlands
- E. Jeffrey Conklin. (2002). *Making Sense of Fragmentary Information: Compendium and the Intelligence Community*
- Prashant Nair. (2003). *Design And Implementation Of An Intranet Driven Radiology Knowledge Bank*. Master Of Science Thesis. University Of Florida
- Marcus Eriksson. (1999). *An ODBC – driver for the mediator database AMOS II*. Linköping University
- J-P. Rosen. (2008). *Experiences in Developing a Typical Web/Database Application:19-21 rue du 8 mai 1945,94110 Arcueil*. France
- Amit, K. (2007). *Securing Web Applications From Application-Level Attack*. B.E., Pune University, India, 2002. M.S, Kent State University, 2007
- Brooke Barnabe. (2007). *The Open Database Connectivity Standard (ODBC)*. [www.cs.siu.edu/~behlman/Classes/cs534/Topic%20Papers/The%20Open%20Database%20Connectivity%20Standard.doc](http://www.cs.siu.edu/~behlman/Classes/cs534/Topic%20Papers/The%20Open%20Database%20Connectivity%20Standard.doc)
- Wei Zheng. (2000). *A Web Based Distributed Medical Record And Imaging Entry And Visualization System*. Degree of Master of Science. University Of Florida
- OpenLink Software. (2003). *Mac OS X Data Source Configuration*: [docs.openlinksw.com/st/osxliteconf.html](http://docs.openlinksw.com/st/osxliteconf.html) - 55k
- Easysoft Limited. (2009). *What's New in SQL Server ODBC Driver Version 1.1* :[www.easysoft.com/products/data\\_access/odbc-sql-server-driver/whats-new.html](http://www.easysoft.com/products/data_access/odbc-sql-server-driver/whats-new.html) - 35k –
- Sun Microsystems, Inc. (2009). *The Java Database Connectivity (JDBC)*. <http://java.sun.com/javase/technologies/database/>
- Shuxin Yuan. (2000). *Development of a Distributed Geoprocessing Service Model*. University of Calgary: URL: <http://www.geomatics.ucalgary.ca/GradTheses.html>
- Mounia Belmamoune. (2003). *A reengineered publication model based on Reuse of a Document Management System And The presentation of an Enterprise Java Application*. University of Amsterdam
- Parag, D., et al. (2004). *Developing a Secure Online Java-based Data Analysis and Visualization Tool*. *Technical Report DU-CS-04-06*. Department of Computer Science, Drexel University. Philadelphia
- B. Fu, S., et . al. (2001). *A Database Federation Platform For Gene Chips And The Human Genome Database*. Massachusetts Institute of Technology, Cambridge, MA USA
- Simon Kainz. (2008). *Data retrieval" and \automatic data post-processing" within iLAP (Laboratory data management, Analysis and Protocol development)*. Master Thesis. nstitute for Genomics and Bioinformatics, Graz University of Technology
- Rakesh Dhaval. (2005). *Gcell A Sub-Cellular Localization Tool*. Master of Sciences thesis. Indiana University
- Awais, R., and Ruzanna, Ch. (2003). *Persistence as an Aspect*. Computing Department, Lancaster University, United Kingdom
- Erdogan, D., et al. (2004). *A Generic Database Web Service*. TOBB Economics and Technology University, Computer Engineering Department, Ankara, Turkey. Georgia State University Department of Computer Science Atlanta, Georgia, USA

- Simon, Zh., *et al.* (2002). *Tele-Electrogastrography*. Department of Electrical and Computer Engineering, University of Calgary, Calgary, Alberta. Department of Electrical and Computer Engineering, University of Alberta, Edmonton, Alberta, Canada T2N 1N4
- Ulrich, U., and Stephanie, T. (2001). Secure Internet-Access to Medical Data. *The European Online Magazine for the IT Professional, Vol. II, No.1, Feb. 2001*.  
<http://www.upgrade-cepis.org>
- Gary Yeung. (1999). Multi-Agent Framework for Immediate Incremental View Maintenance *in Data Warehousing*. B.A.Sc., University of British Columbia
- Barry Cornelius. (1998). Using CORBA and JDBC to produce Three Tier Systems. *SIGPLAN Notices, Volume 33 Issue 4*.  
 Publisher: ACM
- Franklin. (2008). *Network Data Encryption and Integrity for Thin JDBC Clients*:  
[javasight.wordpress.com/2008/08/29/network-data-encryption-and-integrity-for-thin-jdbc-clients/](http://javasight.wordpress.com/2008/08/29/network-data-encryption-and-integrity-for-thin-jdbc-clients/) - 31k
- JDBC Developer's Guide and Reference. (2002). *OCI Driver Support for Encryption and Integrity. Release 2 (9.2). Part No. A96654-01*:  
<http://doc.gold.ac.uk/oracle/9i/java.920/a96654.pdf>
- Gustav Boström. (2003). *Database encryption as an Aspect.KTH, Electrum 213.164 40 Kista. Sweden*:  
[www.isk.kth.se/~gusbo/publications/EncryptionAspectSIGFormatSubmitted.pdf](http://www.isk.kth.se/~gusbo/publications/EncryptionAspectSIGFormatSubmitted.pdf) -
- Microsoft Corporation. (2007). *Microsoft SQL Server 2005 JDBC Driver*:  
<http://msdn.microsoft.com/en-au/data/aa937724.aspx>
- Kenneth, L. (1999). *ADODB: ActiveX Data Objects 2.1*.  
 USA: Microsoft Corporation
- Khalid, E., Neil, E., Daniel, S., Mohamed, Sh. and Gamal N. (2005). *Integrating GIS and MCDM Using COM Technology*. USA and Egypt: The International Arab Journal of Information Technology, Vol. 2, No. 2,
- Ke Wei. (2006). *Structured query language (SQL) related tools*.  
 Ames, Iowa: Iowa State University
- Vorgelegt, Von. (2006). *Enhanced Active Databases for Federated Information Systems*  
 : Universität D'usseldorf
- William Stott. (2007). *Ultrasound Record Archive System For Ukctocs*.  
 Degree of Master of Science, Cranfield University
- Václav Skala and Tom Philip (2004). Utility Computing—Identifying the Applicability Domain and Its Boundaries: West Bohemia. *Journal of .NET Technologies. ISSN 1801-2108 Volume 1, Number 1-3, 2003*: Switzerland, University of Zurich and University of
- Agustinus, B., W. (October, 2005). *Data Broadcasting for Wireless Databases*  
 : Monash University
- Fong, Y., Yee, Ch. And Yong, H., Yi. (2005). *The Sms Based Forecasting System Of Public Transport Arrival Time*: Malaysia. Multimedia University. Faculty Of Information Technology
- Barry Dorrans. (2008). *Data Access using Microsoft ActiveX Data Objects*  
 : The iGroup, a division of Computacenter plc.
- Mayukh Bose. (2004). *What is ADO?*  
<http://www.mayukhbose.com/python/ado/what-is-ado.php>
- S.C. Rennie, *et al.* (2000). *Web Access to Environmental Databases: a Database Query and Presentation System for the UK Environmental Change Network*.  
<http://www.acm.org/conferences/sac/sac2000/Proceed/FinalPapers/WW-01/>
- Steven J. S., *et al* (2004). *A Novel, Web-Driven Continuous Mining Simulator*.  
[www.energy.vt.edu/Publications/2004\\_MPES.pd](http://www.energy.vt.edu/Publications/2004_MPES.pd)
- Ding, L., *et al.* (2007). Virtual Test System Based on Visual C++i. *Electronic Measurement and Instruments, 2007. ICEMI '07. 8th International Conference on Aug. 16 2007- July 18 2007 Page(s):2-276 - 2-278*. Digital Object Identifier 10.1109/ICEMI.2007.4350671

- Microsoft corporation .(2009b). *ActiveX Data Objects (ADO)*.  
[http://msdn.microsoft.com/en-us/library/ms676795\(VS.85\).aspx](http://msdn.microsoft.com/en-us/library/ms676795(VS.85).aspx)
- Ding, L., *et al.* (2007). Virtual Test System Based on Visual C++. *Electronic Measurement and Instruments, 2007. ICEMI '07. 8th International Conference on. Aug. 16 2007- July 18 2007 Page(s):2-276 - 2-278*. Digital Object Identifier 10.1109/ICEMI.2007.4350671
- Julia Norman. (2002). *Evaluation and Implementation of a Persistence Layer for Relational Databases*. Master thesis. Computer Science Department. Uppsala University . S-751 05 Uppsala. Sweden.  
<ftp://ftp.csd.uu.se/pub/papers/masters-theses/0215-norman.pdf>
- Swapna Kodali . (2007). *The Design And Implementation Of An E-Commerce Site For Online Book Sales*. Master degree thesis. Department of Computer and Information Sciences, Indiana University South Bend.  
[http://www.cs.iusb.edu/thesis/SKhodali\\_thesis.pdf](http://www.cs.iusb.edu/thesis/SKhodali_thesis.pdf)
- Atul Adya, *et al.* (2007). Anatomy of the ADO.NET entity framework. *SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data*.  
 Publisher: ACM
- José A. Blakeley et al .( 2006). The ADO.NET entity framework: making the conceptual level real. *SIGMOD Record , Volume 35 Issue 4*.  
 Publisher: ACM
- J.D. Meier *et al.* (2005). *Security Checklist: ADO.NET 2.0*. Microsoft Corporation.  
[http://msdn.microsoft.com/en-us/library/aa480473.aspx#pagck0002\\_sensitivedata](http://msdn.microsoft.com/en-us/library/aa480473.aspx#pagck0002_sensitivedata)
- Objective Interface Systems, Inc. (1996 – 2009). *What is CORBA?*  
<http://www.ois.com/Products/What-is-CORBA.html>
- Desmond, Ch., *et al.* (2001). *Stream Enhancements for the CORBA Event Service*. Ireland. National University of Ireland  
 Publisher: ACM
- Mária, T., *et al.* (2003). *CORBA Based Design and Implementation of Universal Personal Computing*. Canada.  
 Publishers: Kluwer Academic
- Weili Tao, Shikharesh Majumdar. (2003). *Application Level Performance Optimizations for CORBA Based Systems*. CANADA, Ottawa, Carleton University
- Alessio, B., *et al.* (2002). *Use of a CORBA/RMI Gateway: Characterization of Communication Overhead*. Italy, Università di Pisa Dipartimento di Ingegneria dell'Informazione via Diotisalvi, 2 56100 PISA: ACM
- Matteo, P., *et al.* (2003). A Formal Approach for Designing CORBA based Applications. *Transactions on Software Engineering and Methodology (TOSEM), Volume 12 Issue 2*. Italy. Dipartimento di Elettronica e Informazione Dipartimento di Ingegneria dell'Innovazione Politecnico di Milano Università di Lecce: ACM
- E-Kai, Sh., *et al.* (2000). *High Performance Adaptive Middleware for CORBA-Based Systems*. Canada: ACM
- Philip Maechling. (2005). *Distributed Computing Technologies – Selecting an Appropriate Approach*. Web Services Workshop  
 Publisher:UNAVCO/IRIS Joint Workshop
- Darrell, B. *et al.* (2001). Designing an Efficient and Scalable Server-side Asynchrony Model for CORBA. *LCTES '01: Proceedings of the ACM SIGPLAN workshop on Languages, compilers and tools for embedded systems*.  
 Publisher: ACM
- Jianqiang, H., *et al.* (2004 ). *Research and Implementation of CORBA Web Services*.  
[www.springerlink.com/index/DCNYY79MEUXFWJNW](http://www.springerlink.com/index/DCNYY79MEUXFWJNW)
- Liang, G. *et al.* (2000). Java/CORBA technology on a Beowulf cluster. ACM-SE 38: *Proceedings of the 38th annual on Southeast regional conference*.  
 Publisher: ACM

- C. Liebig *et al.* (2000). A publish/subscribe CORBA persistent state service prototype. *Middleware '00: IFIP/ACM International Conference on Distributed systems platforms*. Publisher: Springer-Verlag New York, Inc.
- Bernard, P. *et al.* (1999). Distributed supply chain simulation in a DEVS/CORBA execution environment. *WSC '99: Proceedings of the 31st conference on Winter simulation: Simulation---a bridge to the future - Volume 2 , Volume 2*. Publisher: ACM
- Arnold, B., and Leroy, J. (1998). Distributed simulation modeling: a comparison of HLA, CORBA, and RMI. *WSC '98: Proceedings of the 30th conference on Winter simulation*. Publisher: IEEE Computer Society Press
- Paul, H., and Krishnan, S. (1998). The benefits of CORBA-based network management. *Communications of the ACM , Volume 41 Issue 10*. Publisher: ACM
- Oliver, K., and Alex, J. (2000 ). Using Corba in the Web Operating System. *Distributed Communities on the Web* (pp. 133-141). Heidelberg: Springer Berlin
- Rakman, Ch., *et al* (2006). Design of a security platform for CORBA based application. *Information and Communications Security (pp 98-108)*. Heidelberg: Springer Berlin
- Doland L. (2008). *A Corba-Based Distrubuted And Multi-Threaded Algorith For Finding Related Records In A Large Data Set*. Master's Thesis. Universtiy Of Arkansas.
- Jupitermedia Corporation . (2009). OLAP  
<http://www.webopedia.com/TERM/O/OLAP.html>
- Svetlana Mansmann. (2008). *Extending the OLAP Technology to Handle Non-Conventional and Complex Data: Tech-FAQ*.  
[www.inf.uni-konstanz.de/gk/publications/abstract.html?MaSc08b - 6k -](http://www.inf.uni-konstanz.de/gk/publications/abstract.html?MaSc08b-6k)
- Wikimedia Foundation, Inc . (2008). *online analytical processing*:  
<http://en.wikipedia.org/wiki/OLAP>
- Owen, K., and Daniel, L. (2003). *Attribute Value Reordering for Efficient Hybrid OLAP*. Canada, U. of New Brunswick: ACM
- Ladjel, B., *et al.* (2005). *A Personalization Framework for OLAP Queries*. France, Universit ´e Franc\_ois-Rabelais: ACM
- Karl, H., *et al.* (2000). *Automatically Generating OLAP Schemata from Conceptual Graphical Models*. Germany: ACM
- Thomas, P., and Toby, J. (2002). *Achieving Scalability in OLAP Materialized View Selection*. Michigan, University of Michigan Ann Arbor: ACM
- Fangyan, R., *et al.* (2003). *Spatial Hierarchy and OLAP-Favored Search in Spatial Data Warehouse*. China, IBM China Research Laboratory, Beijing, 10085: ACM
- Surajit, Ch., and Umeshwar, D. (1997). *An Overview of Data Warehousing and OLAP Technology. SIGMOD Record , Volume 26 Issue 1 : ACM*
- Ralph Kimball. (2007). *Dimensional Relational vs. OLAP: The Final Deployment Conundrum*.  
<http://www.intelligententerprise.com/showArticle.jhtml?articleID=199202330>
- Denis and Marie-Chantal (2008). *Design and implementation of an institutional data warehouse from a decision support perspective*. Master's thesis. Universite du Quebec a Trois-Rivieres .Canada
- Mehrdad Jahangiri (2008). *Wolap: Wavelet-Based On-Line Analytical Processing*. Phd Thesis. University Of Southern California.
- Torsten, P., and Günther, P. (2000). Towards OLAP security design — survey and research issues. *DOLAP '00: Proceedings of the 3rd ACM international workshop on Data warehousing and OLAP*. Publisher: ACM
- Ralph Kimball. (2007). *Dimensional Relational vs. OLAP: The Final Deployment Conundrum*. CMP Media LLC . (2007).  
<http://www.intelligententerprise.com/showArticle.jhtml?articleID=199202330>

- S.S. Ahmed (2002). *What is Ole*: Developer Fusion Ltd.  
<http://www.developerfusion.com/article/1892/link-checker/2/>
- wickedwillie (2004). *What is OLE embedding?* : Answerbag, industrious upstart of Demand Media. [http://www.answerbag.com/q\\_view/2783](http://www.answerbag.com/q_view/2783)
- Microsoft Corporation. (2009). *Microsoft OLE DB*.  
[http://msdn.microsoft.com/en-us/library/ms722784\(VS.85\).aspx](http://msdn.microsoft.com/en-us/library/ms722784(VS.85).aspx)
- Raymond P, Kirsch. (1996). *Teaching Ole Automation: A Problem-Based Learning Approach*. La Salle University. Philadelphia: : Acm
- Zhaohui, T., *et al* (2005). Building Data Mining Solutions with OLE DB for DM and XML for Analysis. *SIGMOD Record, Volume 34 Issue 2 : ACM*,
- JOS6A. Blakeley. (1996). *Data Access for the Masses through OLE DB . Microsoft Corporation: ACM*
- Dr. Bruce E. and Kevin S. (2008). Industrial Automation Using OLE. *Compute! Issue 167 / August 1994 / Page 38*: University of Maine Clifton Karnes. Cafe OLE: get ready for Windows 4.0 with OLE 2.0:
- Natalia Sekacheva (2007). *Upgrading of the third generation of the Remote Diagnostic Systems for ABB Marine Propulsion Systems*. Master's thesis. Aapeenranta University Of Technology. Finland.  
<https://oa.doria.fi/bitstream/handle/10024/29900/TMP.objres.639.pdf?sequence=1>
- Campbell, D.G. (1996). Transaction coordination for the new millennium: SQL Server meets OLE transactions. *Data Engineering, 1996. Proceedings of the Twelfth International Conference on. 26 Feb.-1 March 1996 Page(s):162* Digital Object Identifier 10.1109/ICDE.1996.492101
- Anwar, M.R., *et al.*(2004). Human Machine Interface Using OPC (OLE for Process Control). *Engineering, Sciences and Technology, Student Conference On. 30-31 Dec. 2004 Page(s):35 – 40*
- Kruglinski, David J. (1996). *Inside Visual C++*. Microsoft Press, Redmond, WA.
- Blakeley, J.A. (1996). OLE DB: a component DBMS architecture. *Data Engineering, 1996. Proceedings of the Twelfth International Conference on. 26 Feb.-1 March 1996 Page(s):203 – 204. Digital Object Identifier 10.1109/ICDE.1996.492108*
- Wikipedia. (2009). XML: USA. Wikimedia Foundation, Inc.  
<http://en.wikipedia.org/wiki/XML>
- Felix, W., Klaus U. and Holger, M. (2005). Exploiting native XML indexing techniques for XML retrieval in relational database systems. *Proceedings of the 7th annual ACM international workshop on Web information and data management*. Publisher: ACM.
- Andrey, B. And Yannis, P. (March 2005). *Storing and querying XML data using denormalized relational databases* : Springer-Verlag New York, Inc.
- Samuel, S., and Philippe, C. (June 2008). How database and XML can be used to master UML models, an investigation: *Journal of Computing Sciences in Colleges , Volume 23 Issue 6*
- Dunren, Ch., Karl, A. and Tamer, Ö. (2006). Query optimization in XML structured-document databases. *The VLDB Journal , Volume 15 Issue 3* : Springer-Verlag New York, Inc.
- Copyright Jupitermedia Corporation (2009). *XML: Jupitermedia Corporate Info*.  
<http://www.webopedia.com/TERM/X/XML.html>
- Lausesen, S. (2005). *User Interface Design 2005*. Harlow : Pearson/Addison-Wesley, 2005
- Selena Sol (2004). *XML*. The UK Web Design Company (UKWDC). London, UK
- Zachary G. Ives, Alon Y. Halevy Daniel S.Weld. (2002). An XML query engine for network-bound data. *VLDB Journal (2002) 11: 380–402 / Digital Object Identifier (DOI) 10.1007/s00778-002-0078-5*



- Airi, S., and Frank W. (2008). *Requirements for XML Document Database Systems*. University of Waterloo. Canada:  
[www.cs.uwaterloo.ca/~fwtompa/papers/xmldb-desiderata.pdf](http://www.cs.uwaterloo.ca/~fwtompa/papers/xmldb-desiderata.pdf)
- Dimitris, A., and John, S. (2005). XMLNET: Architecture for Cost Effective Network Management Based on XML Technologies. *Journal of Network and Systems Management* : Springer Science+Business Media, Inc. 2005 10.1007/s10922-005-9007-4
- Jung, K., and Hyunchul, K. (2004). Issues in Cache-Answerability for XML Queries on the Web. *Advanced Web Technologies and Applications* (pp 252-257). Heidelberg: Springer Berlin
- Anastasios, I. (2004). Using XML and related standards to support Location Based Services. *SAC '04: Proceedings of the 2004 ACM symposium on Applied computing*. Publisher: ACM
- Win Treese. (2002). XML, web services, and XML. *netWorker* , Volume 6 Issue 3  
 Publisher: ACM
- Jessica Heasley. (2004). Securing XML data. *October 2004 InfoSecCD '04: Proceedings of the 1st annual conference on Information security curriculum development*.  
 Publisher: ACM
- Cheng, J. and Xu, J. (2000). XML and DB2. *Data Engineering, 2000. Proceedings. 16th International Conference on*. March 2000 Page(s):569 - 573 .Digital Object Identifier 10.1109/ICDE.2000.839455
- Muralidhar, K. et al. (2005). Towards an industrial strength SQL/XML infrastructure. *Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on*. 5-8 April 2005 Page(s):991 – 1000. Digital Object Identifier 10.1109/ICDE.2005.144
- Oliveira, E. et al. (2006). Security on MASs with XML Security Specifications. *Database and Expert Systems Applications, 2006. DEXA '06. 17th International Conference on*. 0-0 0 Page(s):5 - 9 Digital Object Identifier 10.1109/DEXA.2006.126
- Kevin Kenan, K. (2006). *Cryptography in the database 2006*.  
 Symantec Press: Linda McCarthy
- Teng, L., and Ping, Y. (2006). A Web Security Solution based on XML Technology. *Communication Technology, 2006. ICCT '06. International Conference on*. 27-30 Nov. 2006 Page(s):1 - 4  
 Digital Object Identifier 10.1109/ICCT.2006.341975
- William M. et al. (2005). Querying and maintaining ordered XML data using relational databases. *ADC '05: Proceedings of the 16th Australasian database conference - Volume 39* , Volume 39 Publisher: Australian Computer Society, Inc.
- Shankar Pal, Mark Fussell, and Irwin Dolobowsky. (2005). *XML Support in Microsoft SQL Server 2005*: Microsoft Corporation.  
[http://msdn.microsoft.com/en-au/library/ms345117.aspx#sql2k5xml\\_topic1](http://msdn.microsoft.com/en-au/library/ms345117.aspx#sql2k5xml_topic1)
- Emerson, O. et al (2006). Security on MASs with XML Security Specifications. *Proceedings of the 17th International Conference on Database and Expert Systems Applications*. (DEXA'06) 0-7695-2641-1/06 \$20.00 © 2006
- Mark Drake. (2007). *New Features in Oracle XML DB for Oracle Database 11g Release 1*. An Oracle White Paper: Oracle Corporation.  
<http://www.oracle.com/technology/products/database/oracle11g/pdf/xml-db-11g-whitepaper.pdf>
- Mitch Ruebush (2005). *Comparing SQL Server 2005 and Oracle 10g as a Database Platform for Microsoft.Net Developer*.  
[download.microsoft.com/download/a/4/7/a47b7b0e-976d-4f49-b15d-f02ade638ebe/SQL2005andOracle10gasDBPlat.doc](http://download.microsoft.com/download/a/4/7/a47b7b0e-976d-4f49-b15d-f02ade638ebe/SQL2005andOracle10gasDBPlat.doc)
- Microsoft Corporation (2009c). *SQL Server and Database Encryption Keys (Database Engine)*. *SQL Server 2008 Books Online*.  
<http://msdn.microsoft.com/en-us/library/bb964742.aspx>
- Opera Software (2005). *Encryption And dbms\_obfuscation\_toolkit*.  
<http://my.opera.com/poddar007/blog/show.dml/47103>

- Charles J. and Brian E. (2008). *Preventing Enterprise Data Leaks at the Source*.  
Sponsored by: Oracle Corporation
- Don Syme (2006). Leveraging .NET meta-programming components from F#: integrated queries and interoperable heterogeneous execution. *ML '06: Proceedings of the 2006 workshop on ML*. Publisher: ACM
- Wikipedia (2009). *.NET Framework*.  
[http://en.wikipedia.org/wiki/.NET\\_Framework](http://en.wikipedia.org/wiki/.NET_Framework)
- Msdn (2009a). *What is the .NET Framework and Visual Studio?*  
<http://msdn.microsoft.com/en-us/vbasic/bb466159.aspx>
- Msdn (2008). *What's New in the .NET Framework Version 3.5 SP1*.  
<http://msdn.microsoft.com/en-us/library/cc713697.aspx>
- Msdn (2009b). *Data Encryption in SQL Server (ADO.NET)*.  
<http://msdn.microsoft.com/en-us/library/bb669072.aspx>
- Bo Feng and Gabriel Wainer (2008). a .NET Remoting-Based Distributed Simulation Approach for DEVS and Cell-DEVS Models. *DS-RT '08: Proceedings of the 2008 12th IEEE/ACM International Symposium on Distributed Simulation and Real-Time Applications - Volume 00*.  
Publisher: IEEE Computer Society
- Viktor Geller, Christelle Scharff (2005). Traditional and more "exotic" .NET languages: VB .NET, J#, C# and SML .NET. *ITiCSE '05: Proceedings of the 10th annual SIGCSE conference on Innovation and technology in computer science education*.  
Publisher: ACM
- Yanhao Zhu, et al. (2005). In-process object-oriented database design for .NET. *SIGITE '05: Proceedings of the 6th conference on Information technology education*.  
Publisher: ACM
- Barwell, et al. (2001). PROFESSIONAL VB.NET. *Limitations of COM (pg. 13)*.  
U.S.A. Wrox Press Ltd.
- Barwell, et al. (2001b). PROFESSIONAL VB.NET. *Limitations of COM (pg. 17-18)*.  
U.S.A. Wrox Press Ltd
- Suresh (2009). *Microsoft .Net vs. J2EE*.  
[www.sis.uncc.edu/~billchu/classes/fall03/itis5166/suresh.ppt](http://www.sis.uncc.edu/~billchu/classes/fall03/itis5166/suresh.ppt)
- Barwell, et al. (2001c). PROFESSIONAL VB.NET. *Limitations of COM (pg. 17-18)*.  
U.S.A. Wrox Press Ltd

## APPENDIX A

```

Oracle SQL*Plus
File Edit Search Options Help
SQL> @18a)UTKrip_BC_Column_E_ok

Please enter your Smartcard PIN #
63

ENTER 'W' FOR WHOLE COLUMN OR 'P' FOR PARTIAL COLUMN ENCRYPTION
P

IF 'W' ENTER THE COLUMN NAME,
IF 'P' ENTER THE ENCRYPTION CONSTRAINT, e.g. ACADEMIC ADVISOR'S NAME
Prof Dr Norbik16

ENTER THE DATA ENCRYPTION MODE? '0' for Base64; '1' for Hex.
1

---> Results of the Block Cipher PARTIAL COLUMN Encryption <---
> Encryption Username - TDP
> Encrypted Acad. Advisor name = Prof Dr Norbik16
> The data encryption mode : Hex
> Number of column data encrypted = 6
> The LIV_AcadAdv - 48267610D8D4475FFE58B807511F2A7F

PL/SQL procedure successfully completed.

SQL> |

```

Figure 1: Partial Column Encryption Process

## Appendix B

```

Oracle SQL*Plus
File Edit Search Options Help
SQL> @18b)UTKrip_BC_Column_D_ok

Please enter your Smartcard PIN #
63

DO YOU WANT TO DECRYPT THE WHOLE COLUMN OR PARTIAL ( ENTER 'W' OR 'P')
P

IF 'W' ENTER THE COLUMN NAME ELSE
IF 'P' ENTER THE ENCRYPTION CONSTRAINT, e.g. ACADEMIC ADVISOR'S NAME
Prof Dr Norbik16

---> Results of the Block Cipher PARTIAL COLUMN Decryption <---
> Decryption Username - TDP
> Decrypted Acad. Advisor Name - Prof Dr Norbik16
> Decryption Mode - Hex
> Acad. Advisor Count - 6
> The Hash_LIV_AcadAdv value - 48267610D8D4475FFE58B807511F2A7F

PL/SQL procedure successfully completed.

SQL>

```

Figure 2: Partial Column Decryption Process.

## Appendix C

```

Oracle SQL *Plus
File Edit Search Options Help
SQL> select * From BCView1 where LIU_Acad_Advisor = 'BA8FD433A9D45668577D758E02BE0FF9';
ACAD_ADVISOR                                STUDENTNAME
-----
LIU_ACAD_ADVISOR                            S          MODE_E
-----
DyweVsBixnHcPzJtB9KE0a                     NORAINI OTHMAN
BA8FD433A9D45668577D758E02BE0FF9          0
DyweVsBixnHcPzJtB9KE0a                     NORH HANDAN
BA8FD433A9D45668577D758E02BE0FF9          0
DyweVsBixnHcPzJtB9KE0a                     SHAH ZIHAN
BA8FD433A9D45668577D758E02BE0FF9          0

ACAD_ADVISOR                                STUDENTNAME
-----
LIU_ACAD_ADVISOR                            S          MODE_E
-----
DyweVsBixnHcPzJtB9KE0a                     SHAHRUL FADZLY
BA8FD433A9D45668577D758E02BE0FF9          0
DyweVsBixnHcPzJtB9KE0a                     SOFFRI YUSSOF
BA8FD433A9D45668577D758E02BE0FF9          0

SQL>

```

Figure 3: Partial Column Encryption Result.

## Appendix D

```

File Edit Search Options Help
SQL> select * From BCView1 where Acad_Advisor = 'Dr ShaFie';
ACAD_ADVISOR                                STUDENTNAME
-----
LIU_ACAD_ADVISOR                            S          MODE_E
-----
Dr ShaFie                                    NORAINI OTHMAN
BA8FD433A9D45668577D758E02BE0FF9          0
Dr ShaFie                                    NORH HANDAN
BA8FD433A9D45668577D758E02BE0FF9          0
Dr ShaFie                                    SHAH ZIHAN
BA8FD433A9D45668577D758E02BE0FF9          0

ACAD_ADVISOR                                STUDENTNAME
-----
LIU_ACAD_ADVISOR                            S          MODE_E
-----
Dr ShaFie                                    SHAHRUL FADZLY
BA8FD433A9D45668577D758E02BE0FF9          0
Dr ShaFie                                    SOFFRI YUSSOF
BA8FD433A9D45668577D758E02BE0FF9          0

SQL>

```

Figure 4: Partial Column Decryption Result.

**Appendix E**

**Figure 5: Project Ghant Chart**

No	Activities	Year 2008												Year 2009											
		Aug				Sep				Oct				Jan				Feb				Mar			
		Weeks				Weeks				Weeks				Weeks				Weeks				Weeks			
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1	Project Preview																								
2	Literature Review																								
3	Studies and Analysis																								
4	Determine the Research Requirements																								
5	System Design																								
6	Implementation and Testing																								
7	Benefits, Discussion and Future Works																								