

TITLE PAGE

**DESIGN AND IMPLEMENTATION OF
STUDENT/DEPARTMENTAL
ADMINISTRATIVE SOFTWARE (SIES),
USING VISUAL BASIC 6.0
{CASE STUDY: DEPARTMENT OF ELECTRICAL AND
COMPUTER ENGINEERING, F.U.T MINNA}**

BY

**SHITTU ABDULRASHEED
(99/ 8371EE)**

**A PROJECT SUBMITTED TO THE DEPARTMENT
OF ELECTRICAL/ COMPUTER ENGINEERING,
SCHOOL OF ENGINEERING AND ENGINEERING
TECHNOLOGY (S.E.E.T), IN PARTIAL
FULFILLMENT OF THE REQUIREMENT FOR THE
AWARD OF BACHELOR DEGREE IN
ENGINEERING AT THE FEDERAL UNIVERSITY
OF TECHNOLOGY, MINNA. NIGERIA.**

NOVEMBER, 2005.

ATTESTATION/DECLARATION

This is to certify that the project designed and reported here was carried out by Shittu Abdulrasheed (99/ 8371EE), of the department of Electrical and Computer Engineering, F.U.T Minna, and that no part of this project was copied from somebody else's work.

Shittu abdulrasheed



07/12/2005

(Student)

Signature

Date

Mrs. Caroline Alenoghena



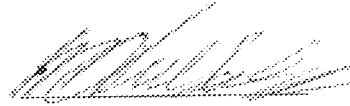
07/12/05

(Supervisor)

Signature

Date

Engr. M.D. Abdullahi



07/12/2005

(H.O.D Electrical/

Signature

Date

Computer Engineering)

ACKNOWLEDGEMENTS

I would like to first and foremost extend my profound gratitude to God Almighty, who has spared my life till now and has allowed me to reach this position in my life. His grace and mercy have been sufficient for me even through the difficult times, and I pray continually for his grace till the end of time.

My gratitude also goes out to my parents Alahji and Hajia M.B. Shittu, who have worked tirelessly through the rigors of life to provide the best for me and the rest of the family. To my father for all the moral and financial advices, I say a big thank you. To my mother for the unending prayers, love, care and affection showered on me. There are no better parents anywhere than the two of you.

I would like to use this opportunity to show my appreciation to my project supervisor, Mrs. Caroline, for her support, understanding and supervision. Others who have contributed immensely to the success of this project.

Also to my siblings, you all were my inspiration. To my friends, Itunu, Mike, Bala, Moh. Chioma, Femi, all others too numerous to mention.

I would never forget Nubwa John and all those that gave me a listening ear and a shoulder to lean on when I had no one around, and all those who believed in me and hence encouraged my work; I will like to say a big thank you.

May Almighty God reward you all and guide you in all your endeavors

DEDICATION

I dedicate this project to Almighty God, and to my beautiful mother Hajia Mairo Shitu.

ABSTRACT

This project represents the study of the computerization/ automation of the administrative jobs in the electrical and computer engineering department using an object oriented software (SIES 1.0), developed with visual basic 6.0 and Structure Query Language (SQL).

The application is a client server application. This means that it needs a Server to work. The server is where the database is stored. Clients are used to connect to this server.

It begins by connecting to the server and after an authorized password the main page can be reached. This program registers and manages students personal information, evaluates student for entry to the next level, defines a curriculum, and prints specified records including summary automation. Hence it eliminates the problem of manual school system which is a laborious task.

TABLE OF CONTENT

Title page
Certification
Acknowledgment
Dedication
Abstract
Table of content

CHAPTER ONE

- 1.0 Introduction to registration and administration
- 1.1 Introduction
- 1.2 Application of computers to departmental registration and students records
- 1.3 Data collection analyses
- 1.4 Objective of data and registration automation
- 1.5 Objective of student evaluation and information system

CHAPTER TWO

- 2.1 Literature review
- 2.2 Theory of object oriented programming and relational management system.
 - 2.2.1 OOP basics
 - 2.2.2 Concepts of OOP
 - 2.2.3 Properties of OOP
- 2.3 OOP languages
 - 2.3.1 Benefits of OOP with visual basics 6.0
- 2.4 Databases
 - 2.4.1 Database models
 - 2.4.2 Microsoft access 2000
- 2.5 Structured query language

CHAPTER THREE

- 3.1 Student's information and evaluation system
 - 3.1.1 Preface
- 3.2 System overview
- 3.3 Client application
 - 3.3.1 Introduction
 - 3.3.2 Getting started
- 3.4 Student's personal information
- 3.5 Add curriculum form
- 3.6 Student's school record form

3.7 Evaluation module

3.8 Summary module

CHAPTER FOUR

4.0 Compiling , packaging and deployment

4.1 System Implementation

4.2 System installation

4.3 System requirement

CHAPTER FIVE

5.1 Conclusion

5.2 Recommendation

5.4 References

Appendix A – Sample source codes

CHAPTER ONE

1.0 INTRODUCTION TO REGISTRATION AND STUDENTS RECORDS

1.1 INTRODUCTION

Registration is the act of enrolling, in other words, registration is the act of keeping research of facts. Therefore, in a school system, it is very important for every student to register before becoming a bonafide student of the school.

Form of registration in higher institution is done twice a session i.e. 1st and 2nd semester but in some other institution like FUT Minna registration is done once in a session which is the very first thing in the first semester.

Good and effective student record must be accurate, precise, complete, relevant, unambiguous and legible for prompt actions also efficient storage, quick retrieval and effective communication of students' record from schools to department and vice – versa cannot be over emphasized. In the provision of a good registration procedure.

All the mentioned attributes of a good record can be best achieved by a computer. Another need for computerizing registration and students' records is to eliminate the antecedent problems that arise in registration procedure.

About a century ago, a spate of invention ushered in the first industrial revolution within a short period of time. many countries became industrialized; the invention of computer amongst many electronic devices has made greater impact on the society than any other device. Computers have strengthened man's power in numerical computation and information processing, thereby increasing the effectiveness of organizations.

Computer is an electro – mechanical device which can accept, process, output and store information according to instructions or programs coded into it without human intervention [1]. They are known to process information in very high speed with accuracy and at the same time, it can store large amount of information.

The times required for computers to sort and process data as well as execute such basic operations as addition and subtraction varies from a few micro-seconds (millionth of a second) to eight Nona-second (billionth of a second), and once the circuit of a computer has been designed, built and fully tested, it is predictable that it will not make mistakes in performing the millions of operation.

Some of the functions of a computer are as follows:

1. Process information according to instructions.
2. Store large amount of data and information (Data is the raw value yet to be processed while information is the processed data).
3. Print text or document.
4. Handle complex and perplexing calculations.
5. Converse with users through terminals.

1.2 APPLICATION OF COMPUTER TO DEPARTMENTAL REGISTRATION AND STUDENTS' RECORDS

Students' record and registration software application is targeted towards achieving 100% data bank/information storage. An easier and effective way of tracking students' record.

Before a student can be registered, he/she must meet the requirement of the proposed department.

Data collected to be saved as students' records include:

Name, state of origin, age, sex, educational qualification, mode of entry, results (JAMB, SSCE, NECO, etc).

The tediousness and boring nature of this work make the use of computers not only necessary but also important. Computers control of records will provide more effective control than any form of manual operation, this is because most of the work to be done manually is repetitive which render it dull for human beings.

In a computerized system, most of the repetition in manual registration and storage of students' data profile is eliminated. A well designed computer system eases the woven job. If computerization of a system is possible, then it has the responsibilities of radical alteration of the products. A computer has been assumed to have an alternative way of encountering what has been affected in the past by manual methods.

A design system can allow a searching procedure in which the user integrates the system until he/she identifies exactly those documents found within it which answers the query.

DATA COLLECTION ANALYSIS

The first step of registration is obtaining the clearance form which tells one to proceed. After all fees have been paid, the collection of course form and fees form which one fills appropriately.

The signing of course form at the department (i.e. electrical/computer engineering) begin the data collection and storage process in the department.

This is when both returning and fresh students indicate the courses taken. Hence, data collection is achieved through registration process.

Most registrations done manually are always inadequate and inefficient in terms of time, legality, accuracy and labour.

- a. Time delay occurs frequently in manual registration.
- b. Legality: The use of handwritten information on the registration sheet in the process of record might lead to misinterpretation or unauthorized deletion or addition. All these may occur due to the handwriting not clear, visible or by omission.
- c. Labour: It is noticed that each session, admitted students increase greatly and the present manual method of registration cannot adequately take care of registering large numbers of students into the department.

This method involves a lot of paper work and method involves a lot of paper and manual filling of paper since the record has to be for a long period, problem of storing them arises.

- d. Data security: The sheets used in students' record collection during registration can easily be tampered with or deliberate rating lost for other personal aims and since the record is large, it cannot easily be detected. Therefore, manual handling of records exposes the record to a lot of risks.
- e. Loss of results: Numerous students from the department of electrical/computer engineering are faced with the problem of missing results and misplaced

continuous assessments due to the use of manual collection and storage of this records.

There is much need to computerize the present registration system because computerized system is more accurate, rapid and exhaustive, compared to manual system which could be error pruned and time consuming rendering officials tired and bored.

To eliminate the many stages involved in the present manual system of registration and retrieval of students' data and results, authorized departmental administrative software is required so that all the needed information can be gotten by simply pressing a key stroke.

1.4 OBJECTIVE OF DATA AND REGISTRATION AUTOMATION

The specific objectives of the study are:

1. To hasten the process of registration and communication of students' data.
2. To reduce stationeries and its costs.
3. To modify the structural adjustments of staff around for more efficient carrying out of duties.
4. To reduce storage space involved in maintaining and keeping of records.
5. To eliminate the problems of missing results and alterations.

For sufficient and efficient data automation, the significant aspects are speed, accuracy, diligent and versatility.

a. **Speed:** Computerization of the system would enable student personal information to be handled quickly and carefully. Processing and output can also be carried out almost instantaneously.

b. **Accuracy:** Computers provide accurate information. Computers can breakdown but it cannot make a computational error, when errors are made in computers, they arise from mistakes in the content of input data.

c. **Diligence:** Computers do not get tired and bored as human agents do and lay aside their task. Computers are systematical, logical and lateral instruction. these instructions allow the computer to carry out some functions unsupervised and spontaneously.

d. **Versatility:** Computers are versatile but only within narrowly defined units. These units are mechanical and desired from the input -- output equipment upon which computer depend but once it has been set up, it can arrange and organize data in ways which would have been difficult normally.

1.5 OBJECTIVE OF STUDENT EVALUATION AND INFORMATION SYSTEM.

In this case we take Federal university of technology, Minna department of electrical and computer engineering to start this operation. F.U.T Minna Department handles different courses. Each has their own curricula to maintain. The following context will explain and will try to provide solution to speed up and increase efficiency in student record management in the said department.

Objectives

1. Speed up transactions on the Department.
2. Eliminate to a minimum value all the problems encountered.
3. Provide a better record management
4. Better record retrieval
5. Provide efficient evaluation of student
6. Create reports for the department

CHAPTER TWO

2.1 LITERATURE REVIEW

Electronic computers have existed since around the middle of this century. The early computers (e.g. ENIAC -- Electronic Numerical Integrator and Calculator in 1940, EDSAC -- Electronic Delay Storage Automatic Computer in 1949, UNIVAC in 1950). Vacuum tubes took over the functions that had previously been performed mechanically [6]. Today, after several generations, miniaturization has brought about the micro-computer in a size compatible with other household furniture and at cost within the reach of many family budgets.

Micro computers are the smallest general processing systems that can execute program instructions to perform a variety of tasks. For such computers to play a part in problem solving, it was necessary that communication be established between them and the users. The bridge of gap in communication was closed with the advent of computer programming languages which translated program written by users into machine codes.

Programming languages are basically of three types: machine language, low level language and high level language. Machine language involves the use of binary codes. Low level languages use symbols in which instructions which corresponded to machine codes were translated using program known as assemblies; high level language is a problem oriented language which is a restricted form of natural language used in programming and uses system software known as Compilers (e.g. FORTRAN, BASIC, COBOL, etc) to translate its codes into a machine language equivalent. While there are many important innovations in software development overall, perhaps the most significant over the past 25

years was the widespread adoption of object-oriented programming (OOP) using high level language.

Object oriented programming (OOP) has many roots, but can be traced back to the 1950s with lisp and the late 1960s with simula. Alan Kay, who carried the term "object oriented programming" from his small talk language is described as the father of (OOP) [1].

He established the fact that programs would be easier to write when they were modeled on things that were easier to understand. He also realized that graphical user interfaces were important as they improved the interaction between people and machines. Object-oriented languages like small talk enabled the widespread adoption of these interfaces and this eventually led to more general software components wired together using pre-written objects and minimal coding. Most importantly, OOP helped lay the foundation for many advances in software engineering, from testing techniques to programming methodologies.

Today, unlike in older ways of coding, the OOP approach uses objects ways of coding, attributes to simplify coding. It offers a powerful model for writing computer software and allows for the analysis and design of an application in terms of entities or objects so that the processes replicate the human thought as closely as possible. In our present world, OOP compilers (e.g. Java, C++ and Microsoft visual basic 6.0) have been used to create all sorts of software ranging from desktop applications to distributed or client-server application. Most of this software has been interfaced with relational databases to achieve desired results. These databases are simply a collection of interrelated tables which store and manage data in tabular formats using a relational database system (RDBMS) [2]. Most importantly, these OOP compilers have been put to use in the

educational sector by using them to create software for school management. A good example of such is the management and information system software.

Management and information system software application package developed using Microsoft's visual basics 6.0 alongside Microsoft access 2002 (an RDBMS) and Microsoft sewer 2000 for automation of the electrical/electronics and computer department of the federal university of technology, Minna, Niger state. It handles administrative jobs (e.g. Departmental registration, course registration, students' evaluation and personal information) in the department and by doing so eases the jobs, manages data efficiently and simplifies administration.

2.2 THEORY OF OBJECT ORIENTED PROGRAMMING AND RELATIONAL MANAGEMENT SYSTEMS

2.2.1 OOP BASICS

In procedure based programming, when designing applications, the focus is on the procedures and functions in the program. The basic algorithm is designed first then implemented in procedural and functions. It can be refined gradually by including more and more details in these functions and procedures. In contract, the focus of object-oriented programming (OOP) is on the data in the application and the methods that manipulate them. It consists of identifying the objects and what is to be done with them as steps to the solution to the problem. These objects have specific roles, communicate and work together to perform the functions of the program. OOP is basically a different way of organizing and structuring programs and is modeled after the real world in which an object is usually made up of other smaller objects.

The concept of OOP listed ahead have a one-to-one correspondence between the way we think about a problem and the implementation [5].

2.2.2 CONCEPTS OF OOP

For object-oriented programming are concepts of objects, classes, properties or attributes, message, methods, interfaces and reusability components [5].

OBJECTS: - An object in OOP represents an entity in the real world. It is designed as a concept or thing with boundaries that is relevant to the problem vehicle, electrical components in a circuit design model, personal files, etc.

An object is also said to be an instance of class [5].

CLASSES: - Objects which have the same properties, common behaviour and common relationships are called classes. Example of classes includes class of polygons, class of animals, etc. Classes help to define the characteristics that object possess and normally consist of interfaces and implemented codes [5].

PROPERTY/ATTRIBUTE: - The characteristics of the object are represented as the variables in a class and referred to as the properties or attributes of the class. For example, in a class of polygons, all objects have vertices and edges as part of their properties or attributes [5].

METHOD: - Refers to an action required of an object or entity when represented in a class. All objections in a class perform certain common actions or operations.

which become functions in the class that defines them. "DRAW", "ERASE" and "MOVE" are examples of methods that are part of the polygon class [5].

MESSAGES: - Software objects interact and communicate with each other using messages. Sending a message to an object cause that object to perform and execute it. The code corresponding to a particular message is known as a "method" [5].

INTERFACES: - An interface is a contract in the form of a collection of method and constant declarations. When a class implements an interface, it promises to implement all of the methods declared in the interface. Most interfaces in object-oriented programming are visible to the user and help in interaction with the program [5].

REUSABILITY OF COMPONENTS: - In all object-oriented languages, programs are broken down into extensible and reusable components. This allows programmes to re-use them later or modify the existing components to create a new one [5].

2.2.3 PROPERTIES OF OOP

Every object-oriented program has four important properties. These are: data, abstraction, inheritance, encapsulation and polymorphism [6].

I. DATA ABSTRACTION: -

This is the process of examining all the available information about an entity to identify properties and methods that is relevant to the application. Grouping of objects into classes is a good example of data abstraction as this causes common definitions to be stored once per class instead of once per instance of a class.

II. INHERITANCE: -

Inheritance is the property that allows the re-use of an existing class to build a new class. The new class created inherits all the behaviour of the original class and is called the sub-class. It is said to be a specialization of the original class.

III. ENCAPSULATION: -

This is a process that allows selective exposing or inching of properties and methods in a class. It involves providing access to an object only through it, messages, while keeping the details private. It is a technique for minimizing interdependences among modules by defining a strict external interface. This way, internal coding can be changed without affecting the interface, so long as the new implementation supports the same (or upwards compatible) external interface. Encapsulation of code helps to protect code from accidental corruption and indicate errors to small sections of code to make them easier to find and fixed.

[6]

2.3 OOP LANGUAGES

Object-oriented languages have gone a long way, today there are several of them in use for many purposes, especially automation. The choice of which language to use depends on the programmer and the features offered by the language [1].

2.3.1 TYPES OF OOP LANGUAGES

The language concepts started from concepts set forth in LOGO and in simula 67 programs. Using these concepts, the first object oriented programming language to be developed was called small talk. It was developed by Alan Kay in the mid 70s. It used run-time binding and a rich class library that could be easily re-used via inheritance. It also had a dynamic development environment which allowed easy refined class definitions. Today, there are almost two dozen major object-oriented languages in use. Some of these are: Ada, Clu, C++, EIFFEL, JAVA, CLOS, F90 and VISUAL BASIC. However, the leading commercial object-oriented languages are C++, JAVA and VISUAL BASIC.

C++ was developed by Bjarne Stroustrup at AT & T Bell laboratories using C language as a base but implementing the principles of object-orientation most effectively to give a powerful language. However, it sacrifices some flexibility and trades off some of the power to re-use classes, in order to remain efficient [1]. JAVA (originally called OAK) was created by James Gosling in 1991 at SUN Microsystems in the USA. Although, it was originally intended for embedded applications, the development of the internet shifted the aims of the project. Today, java is the leading language for web applications.

Visual basic, which was developed from the Procedural BASIC language, was originally developed in 1987 by Alan Cooper under the name of 'Ruby'. At one time,

visual basic could produce code for both disk operating system (DOS) and windows applications. Today, however, DOS is obsolete and visual basic is now a fourth generation programming language used for roughly two-thirds of all business applications programming on personal computers.

As said earlier, all these languages vary in their support of object-oriented concepts. It should however be noted that there is no single language that matches all styles and meets all needs [1].

2.3.1 BENEFITS OF OOP WITH VISUAL BASIC 6.0

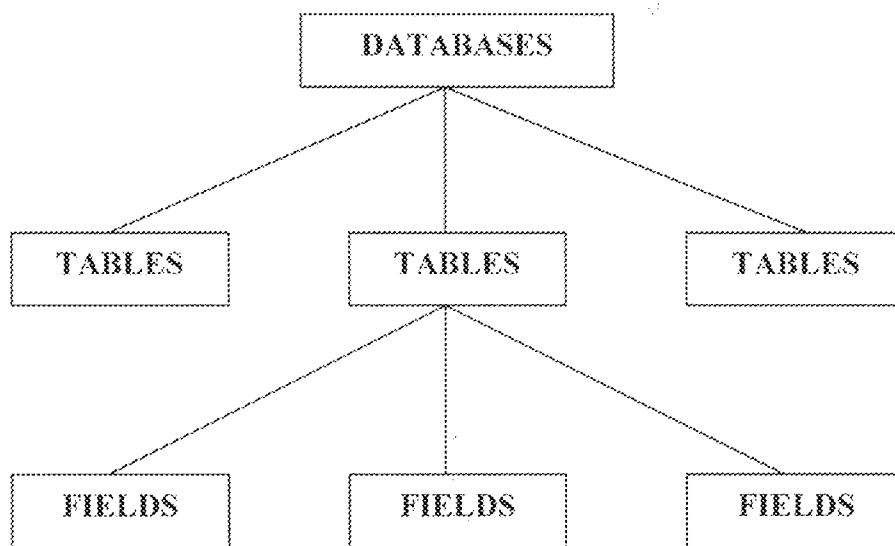
The numerous benefits which VB 6 provides include the following:

- User friendly interface
- It facilitates reuse of code and components and as a result of this, fewer and shorter interactions.
- VB 6 allows for ease of maintenance and enhancement.
- Provides easy access, connection and control of different power databases. (e.g. ORACLE and SQL server) using structured query language.
- Employs tools like "watches", "immediate window" and many others for easy testing, debugging and optimization of programs.
- Contains an integrated data report for creating reports which could be printed out on hard copy or reported to the internet.
- Monitoring of mouse activity and file handling is made easier with VB 6.
- Easy interaction of programs with other windows application.

- Contains an easy to use package and deployment wizard for compiling packaging and deploying software on almost any medium.

2.4 DATABASES

A database is simply a grouping of related information organized for easy processing and retrieval [12]. The actual data in a database is stored in tables. Data in a table is made up of columns and records. The rows contain identically structured pieces of information called records. A record is a collection of values called fields.



[12]

2.4.1 DATABASE MODELS

In the process of evolution, the database was designed on different database models, with each model adding more efficiency to the database. A database model is a description of the data container and a methodology for storing and retrieving data. Before the 1980s, the most commonly used database models were the hierarchical and network systems. In the 80s (and up till now) the 'relational database model' became the rage [11].

2.4.1.1 HIERARCHICAL MODEL

The hierarchical data model organizes data in a tree structure. There is a hierarchy of parent and child data segments. This structure implies that a record can have repeating information, generally in the child data segments. Data in a series of records which have a set of field values attached to it. It collects all the instances of a specific record together as a record type. These record types are the equivalent of tables in the relational model and with the individual records being the equivalent of rows. To create units between these record types, the hierarchical model uses parent-child relationships. These are a 1:N mapping between record types. This is done by using trees, like set theory used in the relational model, "borrowed" from math. For example, an organization might store information about employee, such as name, employee's number, department, salary, etc. The organization might also store information about an employee's children, such as name and date of birth. The employee and children data forms a hierarchy, where the employee and the children data represents the parent segment and the children data represents the child segment.

In a hierarchical database, the parent-child relationship is one to many. This restricts a child segment to having only one parent segment. Hierarchical DBMS were popular from the late 1960, with the introduction of IBM's information management system (IMS) DBMS, through the 1970s [11].

2.4.1.2 NETWORK MODEL

The popularity of the network data model coincided with the popularity of the hierarchical data model. Some data were more naturally modeled with more than one parent

per child. So, the network model permitted the modeling of many-to-many relationships in data. In 1971, the conference on data system languages (CODASYL) formally defined the network model. The basic data modeling construct in the network model is the set construct. A set consists of an owner record type, a set name and a member record type. A member record type can have that role in more than one set, hence the multi-parent concept is supported. An owner record type can also be a member or owner in another set. The data model is a simple network and link and interaction record types (called junction records by IDMS) may exist as well as sets between them. Thus, the complete network of relationships is represented by several pair wise sets; in each set, some (one) record type is owner (at the tail of the network arrow) and one or more record types are members (at the head of the relationship arrow). usually, a set defines a 1:M relationship, although 1:1 is permitted. The CODASYL network model is based on mathematical set theory [11]

2.4.1.3 RELATIONAL MODEL

(RDBMS – Relational database management system)

A database based on the relational model was developed by a mathematician named Dr E. F. Codd at IBM in the late 1960s. He specified a set of 12 rules that have become popular as Codd rules. A database that meets these rules is classified as a ‘true’ complete RDBMS (Relational database management system)

A relational database allows the definition of data structures, storage and retrieval operations and integrity constraints. In such a database, the data and relations between them are organized in tables [11].

PROPERTIES OF RELATIONAL TABLE

- Values are atomic.
- Each row is unique.
- Column values are of the same kind.
- The sequence of column is insignificant.
- The sequence of rows is insignificant.
- Each column has a unique name.

Certain fields may be designated as keys, which mean that searches for specific values of that field will use indexing to speed them up. Where field in two different tables take values from the same set, a joint operation can be preformed to select related records in the two tables by matching values in those fields. Often, but not always, the fields will have the same name in both tables. For example, an "order" table might contain (customer ID, product code) pairs and a "products" table might contain (product code, price) pairs so to calculate a given customer's bill you would sum the prices of all products ordered by that customer by joining on the product code fields of the two tables. This can be extended to joining multiple tables on multiple fields. Because these relationships are only specified at retrieval time, relational databases all classed as dynamic management system. The RELATIONAL database model is based on the relational algebra [11].

2.4.1.4

OBJECT/RELATIONAL MODEL

Object/relational database management systems (ORDBMS) add new object storage capabilities to the relational systems at the core of modern information systems. These new facilities integrate management of traditional fielded data, complex objects such as time series and geospatial data and diverse binary media such as audio, video, images and applets. By encapsulating methods with data structures, an ORDBMS server can execute complex analytical and data manipulation operation to search and transform multimedia and other complex objects.

As an evolutionary technology, the object/relationship (OR) approach has inherited the robust transaction and performance management feature of its relational ancestor and the flexibility of its object-oriented cousin. Databases designers can work with familiar tabular structures and data definition languages (DDLs) while assimilating new object-management possibilities. Query and procedural languages and call interfaces in ORDBMS are familiar: SQL3, vendor procedural languages and ODBC, JDBC and propriety call interfaces are all extensions of RDBMS languages and interfaces. And the leading vendors are of course quite well known: IBM, Informix and Oracle [8].

2.4.2 MICROSOFT ACCESS 2000 – AN RDBMS

Microsoft access 2000, version 9.0 is an RDBMS component of Microsoft office 2000. It comprises of a 'back-end' and a 'front-end'. The 'back-end' part of the application is the one which takes care of storing and retrieving the data. The 'front-end' on the other hand provides a user-interface or a means by which the user can interact with the back-end

data. Most times however, access is simply used as a "back-end" software as its front-end is not as flexible as required [8].

As an RDBMS, access offers many features and responsibilities. Some of these are as follows:

- i. **Large data storage capability:** - Access has the ability to store, manipulate and maintain a maximum of about 65:536 records.
- ii. **Control of data redundancy:** - By storing data in several tables, data repetition or redundancy is reduced in access 2000.
- iii. **Abstraction of data:** - Access includes the actual way data is stored, while providing the user with a conceptual representation of the data.
- iv. **Multiple user support:** Typical of a time RDBMS access provides multi-user support by ensuring that several users can concurrently access data without affecting the speed of data access. It also employs a 'record -- locking' mechanism which ensure that no two user could modify a particular record at the same time.
- v. **Multiple ways of interfacing to the system:** Microsoft access allows a variety of front-end tools to use the database as a back-end. For example, data stored in access can be displayed and manipulated using forms created in FrontPage 2000 or visual basic.

- vi. **Restrict unauthorized access:** - By assigning rights and passwords to user's access is able to implement data security effectively.
- vii. **Backup and recovery:** - In spite of ensuring that the database is secured from unauthorized access as well as invalid entries, there is always the danger of losing the database. To guard the database, access has inbuilt backup and necessary techniques that ensure protection [8].

2.5 STRUCTURED QUERY LANGUAGE

Structured query language (SQL) is used on databases to view and modify only relevant information [12].

SQL is a means of managing data in relational databases. There is a SQL standard. To be able to effectively and efficiently manipulate data, the application accessing the database must implement the structural query language (SQL). SQL is actually a programming structure used to implement the mathematical way of formulating expressions for querying databases. The history of SQL and relational databases traces back to Dr E. F. Codd, an IBM researcher who first published an article on the relational database idea in June 1970. Codd's article started a flurry of research including a major problem at IBM. Part of this project was a database query language named SEQUEL, an acronym for structured English query language. The name was later changed to SQL for legal reasons, but many people still pronounce it SEQUEL; to this day, IBM published many articles in technical journals about its SQL database language and in the late 70s, two other companies were started to develop similar products, which became Oracle and Ingres. By 1985, Oracle

claimed to have over 1000 installations. In the late 1980s and early 90s, SQL products multiplied and became virtually the standard for database management in medium to large organizations [12].

The SQL "language" allows anyone with a computer terminal to access and use relational database. SQL uses about 30 simple "English like commands like open, close, select and update to manipulate the database. For example, the SQL command shown below could be used to select all database records in the electrical engineering department.

```
SELECT * WHERE department = "Engineering"
```

Although, SQL can be used directly by simply typing in commands like this, the SQL language is tricky for non-programmes to learn. However, this student evaluation software allows you to access the database using a standard windows graphical interface.

Student evaluation software translates your mouse clicks and keyboard taps into the SQL language and passes them to Microsoft access [11].

INTERACTION OF VISUAL BASIC 6 WITH DATABASES

Visual basic 6 has several methods of interfacing or connecting to databases. This is done with a efficient approach known as universal data access (UDA). UDA is a philosophy whereby a developer should be able to use one tool to access any type of data from any data sources. This claim is presently being realized through two database access technologies namely; ODBC (Open Database Connectivity) and OLE-DB (Object Linking and Embedding – Database) [4].

ODBC is a standard protocol for accessing relational databases based around SQL. It (ODBC) is an older and less advanced method of connecting to a database. A more advanced way to connect to a database is to use OLE-DB. OLE-DB is a database architecture that provides universal data integration over an enterprise network, from mainframe to desktop, regardless of the data type. It is a more generalized and more efficient strategy for data access than ODBC, because it allows access to more types of data (both relational and non-relational) and is based around Microsoft COM (component object model). Using OLE-DB Microsoft offers an array of techniques for accessing data sources from applications. They include data access object (DAO), remote data objects (RDO) and active X data objects (ADO). For all these, Microsoft Access 2000 supports two data access models. The traditional data access objects (DAO) and active X data object (ADO) [4].

DAO targets the Microsoft Jet database engine to enable quick and easy database programming. Access 2000 is the first version of Access to also support ADO for manipulating Jet databases. Instead of being based on a single database engine, ADO uses a common programming model to deliver access to universal data. It continues the best features of RDO and DAO. It relies on OLE-DB providers for low level links to data sources. It is speculated that OLE-DB technologies will eventually make their ODBC predecessors obsolete much as ADO will replace DAO. Therefore, the connection used for the completion of this student evaluation project (1.0) is ADO.

ADO can be implemented in VB 6 using a data control or using any of its libraries. The data control used is referred to as ADODC (ADO data control).

[4]

CHAPTER THREE

3.1 STUDENT INFORMATION AND EVALUATION SYSTEM

3.1.1 PREFACE

Student Information and Evaluation System (SIES) is a basic School's MIS. The application deals with student personal information, college records, such as their subjects and their grades, summary reports for enrollees, curricula, course and management of the said records. This application is based on schools basic Student Information and Record Keeping and tracking (federal university of technology, Minna). The SIES developed will not aim to totally automate the whole system, but with the aim to automate as it progresses in its development.

I made use of staggered development in implementing the system to easily track down bugs, faults and to easily incorporate needed modules to make the system holistic. The SIES consists of 5 different modules: Student Personal Information (SPI), Student school Record (SSR), Curricula, Student Evaluation System (SE) and Reports Module. The SPI deals with Students Personal Data (e.g. Personal Data Sheet). It keeps track on student information to be kept and will be easily retrieved when needed. Curricula also, keep the curriculum information for a certain school year. SE, is a basic enrollment procedure for every school. Students are first evaluated for their subjects before they can enroll. The role of this SE is to reduce the time for Evaluation by easily comparing Student's college record to the curriculum where they belong. This will facilitate a better and more efficient Enrollment procedure. Lastly, the Reports module deals with summary reports on enrollees for the whole department, for each course and year level.

3.2 SYSTEM OVERVIEW

Due to some circumstances, I chose incremental development model. This software engineering model enables one to integrate modifications, check and implement bugs in the system. This will also allow the department to easily monitor user requests for full system performance. In concept, the system runs on a network based/managed organization. It requires a server to store the records, a back up server incase the server goes down, and client hosts, who will serve as access end points for the server. The server holds the key for the system access. It makes use of SQL Server authentication/User logins. A client/User can access the system as long as the System Administrator grants access to his account, or has given an account.

The server of course, has full access to the entire database. The set up for a network based system allows user to access the system any where as long as the computer is connected in their network group. Development interface used is Microsoft's Visual Basic 6.0. The choice is based on VB6s Rapid Application Development Technique and of course. This Choice is influenced by the fact that VB6 database flexibility would be a plus factor on its network operation. The following data describes the table definition of the SIES:

Tables

Courses

	Column Name	Datatype	Length	Precision	Scale	Allow Nulls
?	Course	nvarchar	15	0	0	<input type="checkbox"/>
	Department	nvarchar	10	0	0	<input checked="" type="checkbox"/>
	Description	nvarchar	100	0	0	<input checked="" type="checkbox"/>

Curriculum

	Column Name	Datatype	Length	Precision	Scale	Allow Nulls
?	SY	nvarchar	10	0	0	<input type="checkbox"/>
?	Sem	nvarchar	3	0	0	<input type="checkbox"/>
?	Course	nvarchar	15	0	0	<input type="checkbox"/>
?	Yr	nvarchar	1	0	0	<input type="checkbox"/>
?	SC	nvarchar	15	0	0	<input type="checkbox"/>
	Description	nvarchar	60	0	0	<input checked="" type="checkbox"/>
	units	real	4	24	0	<input checked="" type="checkbox"/>
	Prerequisites	nvarchar	20	0	0	<input checked="" type="checkbox"/>

Names

	Column Name	Datatype	Length	Precision	Scale	Allow Nulls
	idnum	nvarchar	10	0	0	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	lnam	nvarchar	30	0	0	<input type="checkbox"/>
<input checked="" type="checkbox"/>	fnam	nvarchar	30	0	0	<input type="checkbox"/>
<input checked="" type="checkbox"/>	mnam	nvarchar	30	0	0	<input type="checkbox"/>
	CS	nvarchar	1	0	0	<input checked="" type="checkbox"/>
	Age	smallint	2	5	0	<input checked="" type="checkbox"/>
	College	nvarchar	15	0	0	<input checked="" type="checkbox"/>
	Course	nvarchar	15	0	0	<input checked="" type="checkbox"/>
	Major	nvarchar	20	0	0	<input checked="" type="checkbox"/>
	Minor	nvarchar	15	0	0	<input checked="" type="checkbox"/>
	Bday	smalldatetime	4	0	0	<input checked="" type="checkbox"/>
	Placebirth	nvarchar	255	0	0	<input checked="" type="checkbox"/>
	Sex	nvarchar	1	0	0	<input checked="" type="checkbox"/>
	Nationality	nvarchar	15	0	0	<input checked="" type="checkbox"/>
	HAddress	nvarchar	255	0	0	<input checked="" type="checkbox"/>
	Guardian	nvarchar	75	0	0	<input checked="" type="checkbox"/>
	rlntship	nvarchar	10	0	0	<input checked="" type="checkbox"/>
	Raddress	nvarchar	255	0	0	<input checked="" type="checkbox"/>
	Occupation	nvarchar	25	0	0	<input checked="" type="checkbox"/>
	nmemther	nvarchar	75	0	0	<input checked="" type="checkbox"/>
	moccup	nvarchar	25	0	0	<input checked="" type="checkbox"/>
	nmefther	nvarchar	75	0	0	<input checked="" type="checkbox"/>
	foccup	nvarchar	25	0	0	<input checked="" type="checkbox"/>
	addr	nvarchar	255	0	0	<input checked="" type="checkbox"/>
	Income	nvarchar	20	0	0	<input checked="" type="checkbox"/>
	rel	nvarchar	25	0	0	<input checked="" type="checkbox"/>
	boarding	nvarchar	255	0	0	<input checked="" type="checkbox"/>
	nmelanlord	nvarchar	25	0	0	<input checked="" type="checkbox"/>
	scondary	nvarchar	25	0	0	<input checked="" type="checkbox"/>
	yrgruated	nvarchar	4	0	0	<input checked="" type="checkbox"/>
	Eschladd	nvarchar	255	0	0	<input checked="" type="checkbox"/>
	coll	nvarchar	30	0	0	<input checked="" type="checkbox"/>
	yrgrad	nvarchar	4	0	0	<input checked="" type="checkbox"/>
	Cscladd	nvarchar	255	0	0	<input checked="" type="checkbox"/>

Relationships

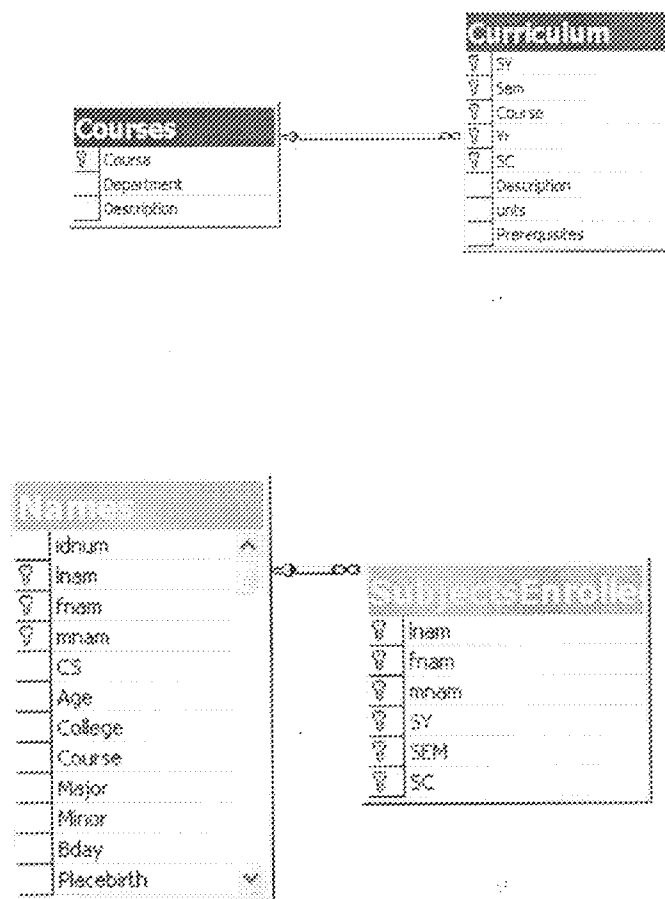


Fig:1.b

3.3 CLIENT APPLICATION

3.3.1 INTRODUCTION

The system is composed of 4 different modules namely Student Personal Information (SPI), Student school Record (SSR), Curricula, Student Evaluation System

(SE) and Reports Module. Each module will be taken here to explain how they work and operate over the client side application.

3.3.2 GETTING STARTED

The system provides a simple but tight system security (Security issues are discussed at the server side, not to be discussed to users!). All user accounts are stored and managed by the Administrator in the SQL SERVER (version 7). Upon loading, user is asked to enter the server you want to connect and a user name and a password. Server name is the name of the server where your database is stored. With the server, you can now log in if you have your user name and account present in the server itself.

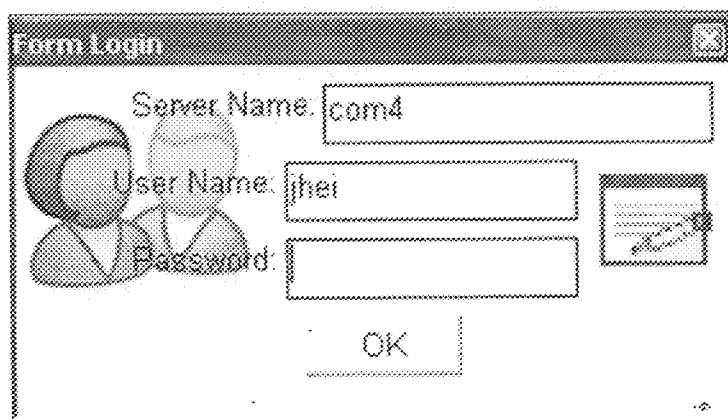


Figure 1.1

Supply all the fields and click ok. The application will resolve the server. If the server is not present, you can't load the system. A correct server and user name and password will continue loading the system.

MANAGEMENT INFORMATION SYSTEM

School
Department

APPLICATION MODULES

STUDENT PIS	CURRICULA	COURSES	REPORTS	EVALUATE STUDENTS
----------------	-----------	---------	---------	----------------------

MISCELLANEOUS

SYSTEM INFO

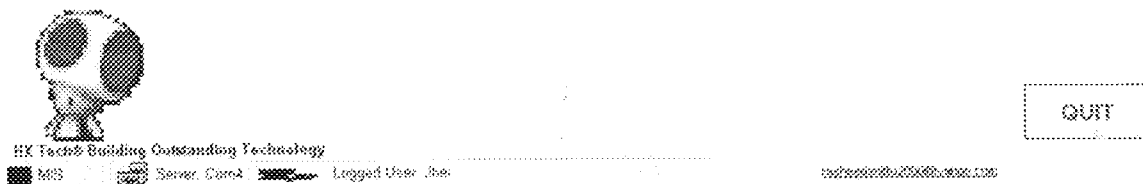


Figure 1.2

Figure 1.2 is the main form of the Application. It consists of three parts, the Modules, Miscellaneous and the Exit portion.

Modules consist of 5 modules (modifications are made the picture may not represent the real system UI). Student PIS is for SPI, SCR and Evaluation, Curricula, is for Curriculum management, courses, for course management and reports for report management.

MANAGEMENT INFORMATION SYSTEM

School
Department

APPLICATION MODULES



MISCELLANEOUS



Evaluation Module. This module is moved on the SPIS (implemented)

Reports Module. This button lets you create reports. When you are logged, you can access this, depending on your rights in the server.

SPI, SCR, SES Module. This button lets you access the SPI, SES and SCR. When you are logged, you can access this, depending on your rights in the server.

Curricula Module. This button lets you access the Curricula. When you are logged, you can access this, depending on your rights in the server.

QUIT

MS Server Logged User: jhu

Company: Formacion CIB

Exit the System

Fig:1.2.1

3.4 STUDENT'S PERSONAL INFORMATION (SPI)

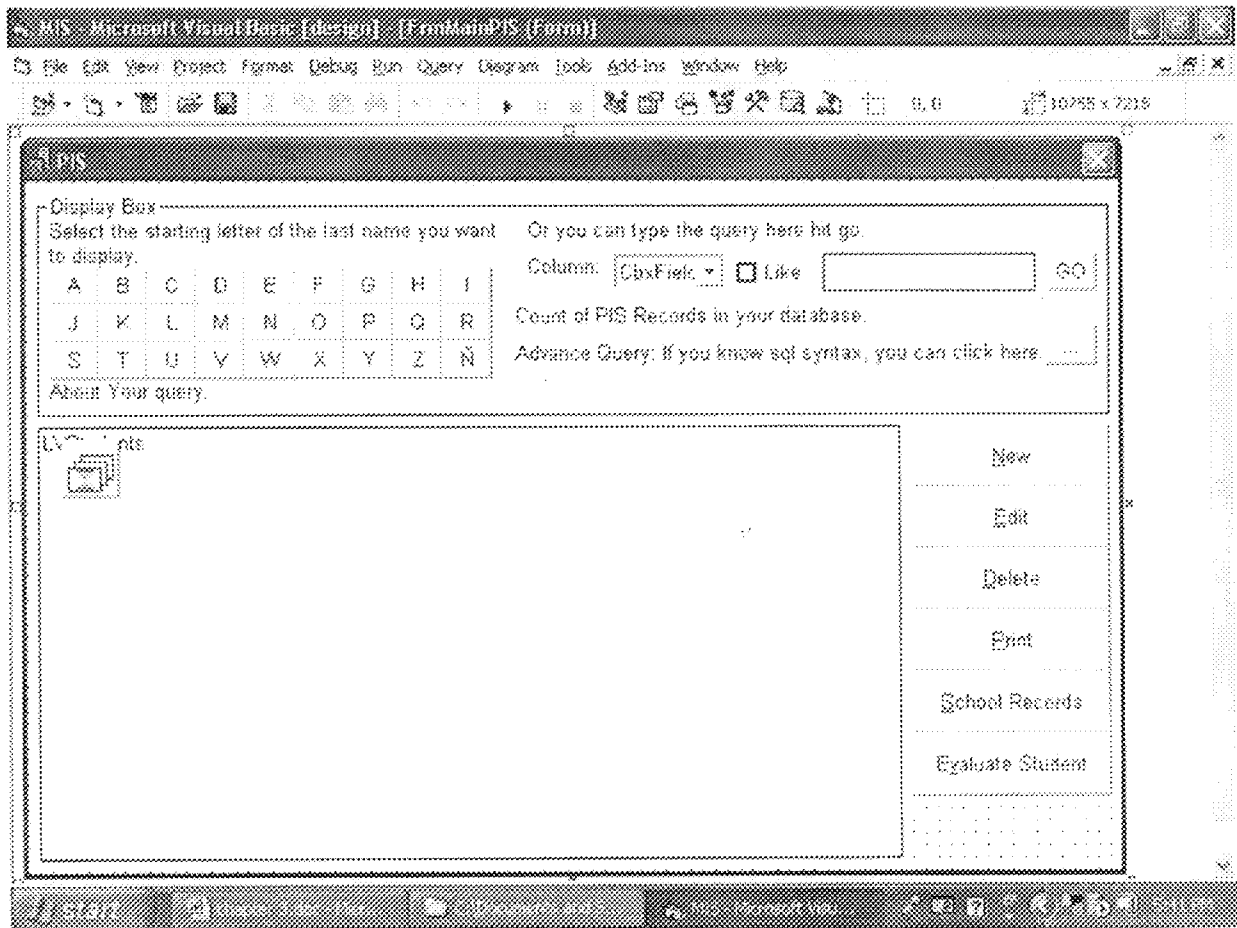


Figure 1.3

Figure 1.3 displays the main form for SPI module. The form contains 3 portion, Query portion, Record display and command groups.

Query Portion – This portion allow the user to select records present in your database. There are 2 parts of this portion, the A-Z buttons and Query part. A-Z button, as its name requires, when trying to click one button, all records starting with the corresponding letter will be displayed in the Record display box. By default, A is clicked when you load this

module. The query box is a more advance record displaying part. Users are able to type a portion or an approximate value or you may type the entire word to search. But first, select the Field you want your query is to be compared. Approximate comparison must click the like box. This will enable you to query approximate matches. You should also add wild cards to enable approximate query (%). Wild cards must be used (e.g. jo% will query John, Jonna, Joseph, Johnattan ... all records that starts with jo). Basically, the approximate matching is slower that exact matching. To continue, click go. This will display (if something matches your query) records matching your query.

Display portion – This portion will display records queried. The display portion displays ID Number, Last Name, First Name and Middle Name.

Command Group – This contains the available commands in the said module. It includes Adding Records, Editing Records, Deleting, Printing, school Record (Point to SSR Module) and Evaluating (Point to Evaluation Module).

Adding Records – Adds new Student Personal Information Record to your system (At least if you are permitted to do so.).

Editing Records – Permits you to edit an existing Record (if you are permitted by the server). This requires a current record. You must select a record/highlight a record in the display record portion box and click Edit or double click or right click and click edit.

Deleting Records – Deleting records is one of the most critical part of a database program. Deleting a Student Record (Personal Information) will also delete all its school records. You can only delete if you are permitted by the server. But before you can delete, you are asked to enter a password (security purpose). If you did not supply the correct password, you are not allowed to delete the record.

Printing – This will print the Personal Information of the selected student (The one highlighted in the Display record portion).

3.5 ADD CURRICULUM FORM

The screenshot shows a window titled "Curriculum" with the following elements:

- School Year Information:** A section containing two dropdown menus: "School Year" (with a highlighted selection) and "Semester" (set to "Sam").
- Course:** A section containing a dropdown menu for course selection, a dropdown menu for "1", a checkbox labeled "Show All Year Level", and a "Load Records" button.
- Table:** A table with the following headers: "YR", "Subject Code", "Description", "Units", and "Pre". The table body is currently empty.
- Buttons:** "Add" and "Delete" buttons are located at the bottom left of the window.

Figure 1.4

Figure 1.4 is the Form for Curriculum adding. This Form contains 3 commands, Load, Add and Delete. You may not see the Edit portion. Editing is done by double clicking the Display Record portion (if there is a record to be edited). The form contains 3 parts, School Year Information, Course Portion and Display Record Portion. School Year Information contains the School Year and Semester you want to load in your Display Record Portion. The Course Portion contains the Course and Year level you want to display (if you want to display all records for a certain school year, semester and course, check the show all year level box). Click the Load Subjects button to load the records.

The active School year, semester, course and year level is the only active record for adding (e.g. SY 2000-2001, SEM -- 2nd, Course -- electrical/computer, Yr -- 1, here you can only add subjects here).

Commands available are:

Load Subjects -- Loads the subjects for the specified school year, semester, course and (optional) year level.

Add -- Adds a new subject for a specified school year, semester, course and (optional) year level.

Delete -- Deletes a subject for a specified school year, semester, course and (optional) year level.

Edit (not shown) -- to activate, double click a record in the display box. Edits a record.

3.6 STUDENTS SCHOOL RECORD FORM

This form is activated by simply clicking the school Record button in your SPI Form (it will only pop up if you have selected a student in the display portion). There are 5 portions found in this form namely, School year, Course, Display, Commands and Summary portion.

School Year -- This portion talks about the school year and semester you want to view for the selected student. Just select here the necessary information then proceed at the course portion.

Course -- As its name implies, it talks about the course and year. After setting the School Year and Course you can now click load subjects to load all the records found in your display portion.

Display -- Display what is being queried or selected (SY, SEM, COURSE, YR).

Commands -- The commands present here are:

New -- Adds a new subject on the selected School Year, Semester, Course and Year. You can click the new button or right click the display portion and click new. A dialog box appears for adding. Fill up the form and click ok.

Edit -- Unless you have selected a subject in the display portion, you cannot edit. Edit enables you to edit the current subject (usually use to update grades).

Delete – Deletes the selected subject (only if you are permitted).

Print – prints the college record of the selected student (not yet implemented, you may want to code it your self).

The move Subject to another School Year and Semester is an invisible command, you can only see this when you right click the display portion. The selected subject will be transferred to a certain school year and semester, depending on your settings. A wizard will actually help you do this.

3.7 EVALUATION MODULE

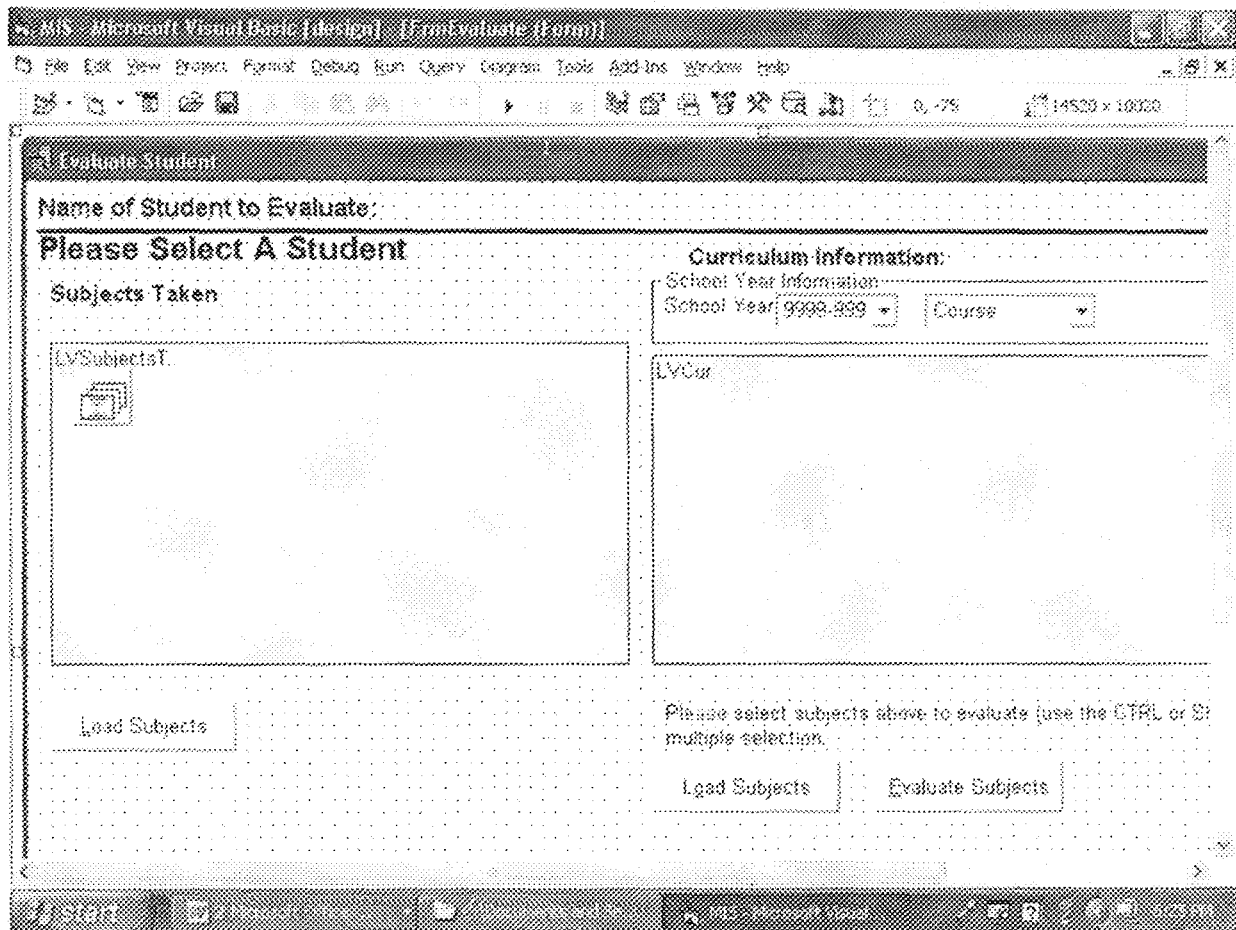


Figure 1.7

Figure 1.7 represents the Evaluation Module. It contains 2 display portion, the Subjects taken by the selected student and the Curriculum you want to use for evaluation.

Evaluation is the process of checking subjects for student if he/she is qualified to take the subjects or not. This depends on the pre requisites of the subjects being evaluated. An evaluation report is generated after evaluation (not printed just a window summary).

You can activate this form by selecting a student in your SPI module form and click the evaluate student button.

The load Subject (Left side) is used to load subjects of the selected student (But automatically if you have loaded the curriculum and hit the Evaluate subjects, this will be loaded, and basically it is not necessary in processing. It is just used to show the user a visual representation of the records being taken by the student.).

To complete evaluation, you must load a curriculum in which you want to evaluate subjects to be taken by the student. Select a school year and course and hit the load subjects button just below the curriculum display portion. After that, all the records of the said curriculum and course will appear. Select the subjects you want to evaluate (Multiple select is permitted, hold the control key and select records). After selecting the necessary records, click the Evaluate subjects and the system will process it. An evaluation report will appear. You can determine which subjects are allowed to be taken by the student. Note that the evaluation report is not a printed output.

3.8 SUMMARY REPORT MODULE

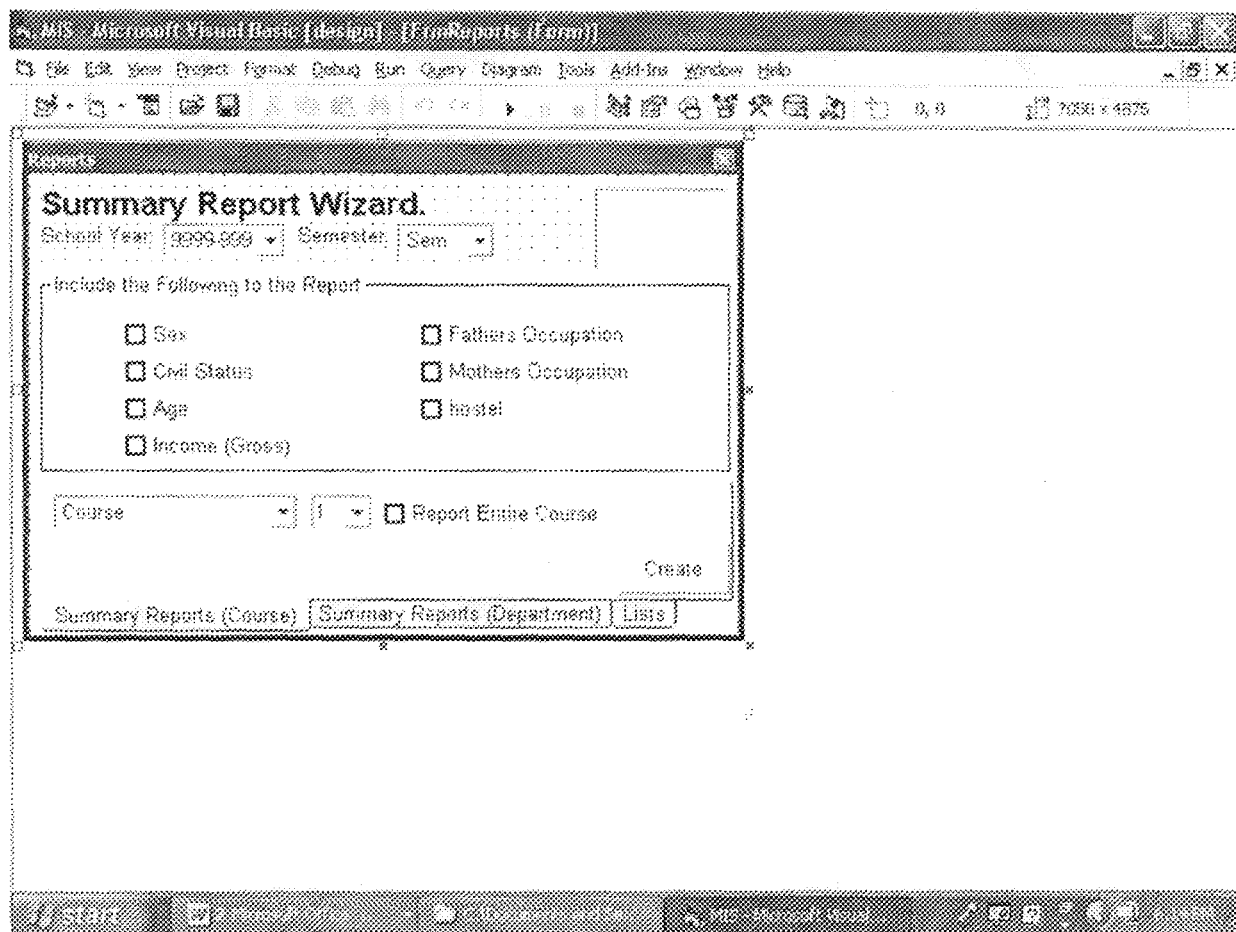


Figure 1.8

Figure 1.8 represents the Summary report module. This will generate an HTML formatted report for certain school year and semester. You can choose the data to be included in the report (e.g. sex, civil status, age etc.). You can generate report by course and year level and of course, with the entire department.

It must be clear the difference between curriculum and subjects offered in a school year. A curriculum is set of subjects prepared every year to be followed by students. This

does not mean that all the subjects are to be offered that school year. Take note that each student follows their own curriculum, thus, subjects offered in a school year are mixture of different subjects from different curricula. A good example is that, A fourth year subjects on Curriculum Year 2001-2002 will be offered on School Year 2005-2006, and first year subjects for curriculum year 2005-2006 are offered in school year 2005-2006. With this complexity, I have separated a table for Subjects offered and Curricula. Subjects offered are used for adding subjects for school records and curricula is used for evaluation and of course, for adding subjects to subjects offered in a certain school year.

The module is still considered as add on module for curricula, but it is separated. The same functionalities, techniques and procedures are used here just like in your curricula module.

That's how easy it is to navigate the system. You can now exit it by clicking the quit button in your main window.

CHAPTER FOUR

4.0 COMPILING, PACKAGING AND DEPLOYMENT

After all the programs have been written and tested they undergo three final processes. The first of these, which is done in the visual basic integrated desktop environment (VB IDE), is the compilation into executables (i.e EXE). These enable the program to run as an executable on any microcomputer.

The next two steps (packaging and deployment) can be performed with an add-on in the VB IDE or with a standalone component of the visual studio suite called 'packaging and deployment wizard'. Packaging is the act of creating a package of one or more cabinet files that contain the project files, dynamic link libraries (DLLs), and any other files the user must have to install in order to run the application. This creates the installation program for the software.

After packaging, the package has to be deployed using the same wizard. Deployment is the process of sending application components and related file from one computer to another or from one location to the other so that the components may be available to other users. The components of the SIES program in this case is packaged as cabinet files and then deployed on a compact disc for distribution. This is the final stage after which the software is ready for use.

4.1 SYSTEM IMPLEMENTATION

4.1.2 IMPLEMENTATION.

Program software

This System was developed using 32-bit version of MS visual Basic Programming language version 6.0, a window-based programming language.

4.2 SYSTEM INSTALLATION

Operating Environment

The operating environment used is windows XP but the software can run on Windows 95 and above operating system

4.2.1 USER MANUAL GUIDE

In using the student information and evaluation Software program, the visual basic environment is not necessarily required with the use of the deployment package which can be installed by simply following the instructions on the screen. Automatically a desktop short cut appears which can be used to operate the software.

4.2.2 STAFF TRAINING

Each staff required to use this program must be thoroughly trained. The access levels for all staffs have to be thoroughly understood and the administrator of the server is required to give all required passwords.

4.2.3 Scope and Limitations

The said project is not as ambitious as it is. It will only try to eliminate some difficulties of the existing system. Basically it will not touch faculty records, schedules. It will mainly focus on Student Personal Information, Student school Record Maintenance, Evaluation, Curricula storage and management and Departmental Reports.

4.3 SYSTEM REQUIREMENT

The SEIS requires the following:

Hardware:

(Client)

Monitor with screen resolution of 1024x728

Compatible Mouse and keyboard

Any Processor with a speed of 1.5 GHz or higher recommended

128mb of RAM

Video card with memory of 32mb or higher recommended

Network Interface card

Uninterrupted Power Supply

(Server)

Monitor with screen resolution of 1024x728

Compatible Mouse and keyboard

Any Processor with a speed of 1.5 GHz or higher recommended (Dual Processor Recommended)

128mb of RAM (512 recommended)

Video card with memory of 32mb or higher recommended

Network Interface card

Uninterrupted Power Supply

Peripherals:

Network Cables

Switch/Hub

RAID Devices

CD-ROM

Software:

(Client)

Any Windows Operating System (Win 2000, XP Professional Recommended)

SQL Server Version 7.0 (Standard Edition)

Internet Explorer 5.1 or higher (Any web browser)

Microsoft Windows Common Controls must be installed

Microsoft Office (Any)

(Server and Back Unit)

Any Microsoft Windows Server Operating System (Advanced Server 2000 recommended)

SQL Server Version 7.0 (Standard Edition) or higher

Internet Explorer 5.1 or higher (Any web browser)

Microsoft Windows Common Controls must be installed

Microsoft Office (Any)

Microsoft Visual Studio 6.0 (Enterprise Recommended)

Network Utility Program

Back up utility program

CHAPTER FIVE

5.1 CONCLUSION

This work describes the students' information and evaluation management Administrative software whose function is to keep track of all student and administrative work.

The manual procedure was analyzed and the disadvantages enumerated. The system was designed to be menu-driven giving rise to easy information retrieval. It also has a modular design in order to make future expansion easy.

The system was subjected to several departmental operations, the system will increase the efficiency and accuracy of tracking down students' academic records, registration details and evaluating students for entry to the next level based on departmental registration.

5.2 RECOMMENDATIONS

For the purpose of future developments the following the following issues should be of concern.

- Security- Due to nature of the system, necessary steps should be taken to enhance its security features, especially with user passwords within the staffs.
- Traffic- Proper and quality hardware facilities should be used for fast movement and retrieval of data over networks.
- More modules can be added, such as online connection to database to allow students register online while given an access level password. This in turn can be used as funds generator for the department.

REFERENCES

- [1] Francis Shield; Computers and Programming, Shaum's outline series, McGraw-hill Book Company, Singapore. International Edition; 1993, pp 20 – 86.
- [2] Ravi R. & Skakhar S.; Software Engineering Modules 6 & 7, First Edition, Software Project Management and Quality Assurance (SPM & SQA), APTECH LIMITED, India; 1997, pp 39 – 70.
- [3] John Smiley; The History of Visual Basics, Revised Edition, Smiley and Associates Inc., USA, 2000, pp 115 – 247.
- [4] Gary Cornell; The Visual Basic 6 from the Ground Up, First Edition, Osborne/McGraw-hill, USA; 1998, pp 9 – 120.
- [5] Design Team, H. O., Object Oriented Programming with C++, First Edition, APTECH Limited India; 1999, pp 24 – 57.
- [6] Design Team; "Object Oriented Analysis and Design with Unified Modeling Language", First Edition, APTECH Worldwide, USA; 2003, pp 6 – 39.
- [7] Design Team; "Visual Basic 6.0 Desktop", First Edition, APTECH Worldwide, USA; 2002, pp 3 – 17.
- [8] Design Team; Database Design with MS Access 2000, First Edition, APTECH Worldwide, USA; 2000, pp 16 – 44.
- [9] Microsoft Corporation; Microsoft Digital Network Library (MSDN), 2001, Revised Edition, Microsoft Corporation, USA; pp 115 – 219.

- [10] Microsoft Corporation, www.microsoft.com, Microsoft Developers Network Library, Database Model Home Page.
- [11] Powered by Google, www.howstuffswork.com, UNIX Space, Database Models, Home Page.
- [12] S. Walla; Indosoft Visual Basics Training Software, an Ss 1 Company, India, (1998), Structures Query Language (SQL) Link.

Sample codes used in the module

Codes to log in

```
Private Sub cmdok_Click()
Dim x As Boolean
x = connect(txtuser.Text, txtpass.Text, Txtserver.Text)
If x = False Then
    MsgBox "Invalid user...", vbCritical, "Error"
Else
    MyLoginPass = txtpass.Text
    MainForm.ssbr.Panels(2).Text = "Server: " & StrConv(Txtserver.Text, vbProperCase)
    MainForm.ssbr.Panels(3).Text = "Logged User: " & StrConv(txtuser.Text,
vbProperCase)
    MainForm.Show
    Unload Me
End If
End Sub
```

```
Private Sub Form_Load()
retriveRegs
End Sub
```

```
Private Function retriveRegs()
    Dim server As String, user As String
    server = GetSetting("CSUMIS", "INFO", "Server", "SQL SERVER NAME")
    user = GetSetting("CSUMIS", "INFO", "User", "User")
    Txtserver.Text = server
    txtuser.Text = user
End Function
```

```
Private Sub txtpass_GotFocus()
    SendKeys "{HOME}+{END}"
End Sub
```

```
Private Sub txtpass_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then cmdok_Click
End Sub
```

```
Private Sub Txtserver_GotFocus()
    SendKeys "{HOME}+{END}"
End Sub
```

```
Private Sub Txtserver_LostFocus()
    SaveSetting "CSUMIS", "INFO", "Server", Txtserver.Text
End Sub
```

```
Private Sub txtuser_GotFocus()
```

```
SendKeys "{HOME}+{END}"  
End Sub
```

```
Private Sub txtuser_LostFocus()  
SaveSetting "CSUMIS", "INFO", "User", txtuser.Text  
End Sub
```

Codes used in the evaluation module

Private Sub BtnEval_Click()

```
If LVCur.SelectedItem Is Nothing Then  
MsgBox "There are no selected subjects to evaluate. Please Select from the list  
above.", vbInformation, "Evaluation"  
Exit Sub  
End If  
'try clicking the button load  
BtnLoadSub_Click  
'continue here  
Dim i As Integer, msg As String  
  
For i = 1 To LVCur.ListItems.Count  
  
If LVCur.ListItems(i).Selected = True Then  
Evaluate LVCur.ListItems(i).SubItems(2), LVCur.ListItems(i).SubItems(5),  
FrmEvaluationRep.Txt1  
End If  
Next  
'partition the message  
FrmEvaluationRep.Caption = "Evaluation Result for " & LblName.Caption  
FrmEvaluationRep.Show 1  
End Sub
```

```
Private Sub BtnLoadcur_Click()  
'Try loading the Subjects  
LoadCurriculum  
End Sub
```

```
Private Sub BtnLoadSub_Click()  
Dim spl  
spl = Split(LblName.Caption, "|", , vbTextCompare)  
Idnum = spl(1)  
Lnam = spl(2)  
Fnam = spl(3)  
Mnam = spl(4)  
'Set the sy and sem  
Dim msg As String
```

```

msg = "Select * from SubjectsEnrolled where lnam = '" & Lnam & "' and fnam = '" &
Fnam & "' and mnam = '" & Mnam & "' Order by SY,SEM,COURSE,YR"
SetSelectedStudSub msg
LoadALLtoLV LVSubjectsTaken
End Sub

```

```

Private Function LoadCurriculum()
LVCur.ListItems.Clear
On Error GoTo Erb
Dim rs As New Recordset
With rs
    Dim msg As String
    msg = "Select * from Curriculum where SY = '" & TxtSy.Text & "' and course = '" &
TxtCourse.Text & "' order by Yr, sem asc"
    .Open msg, DE.Con, adOpenDynamic, adLockOptimistic
    If .RecordCount <> 0 Then
        Do Until .EOF
            LVCur.ListItems.Add .AbsolutePosition, , .Fields("Yr").Value, 3, 3
            LVCur.ListItems(.AbsolutePosition).SubItems(1) = .Fields("Sem").Value
            LVCur.ListItems(.AbsolutePosition).SubItems(2) = .Fields("SC").Value
            LVCur.ListItems(.AbsolutePosition).SubItems(3) = .Fields("Description").Value
            LVCur.ListItems(.AbsolutePosition).SubItems(4) = .Fields("Unts").Value
            LVCur.ListItems(.AbsolutePosition).SubItems(5) = .Fields("Prerequisites").Value
            .MoveNext
        Loop
    End If
End With
Set rs = Nothing
Exit Function
Erb:
MsgBox "Error: " & Err.Description, vbCritical, "Error"
Set rs = Nothing
LVCur.ListItems.Clear
End Function

```

```

Private Function LoadALLtoLV(lv As ListView)
lv.ListItems.Clear
With DE.rsSubjectsEnrolled
    If .RecordCount > 0 Then
        Do Until .EOF
            If UCase(Trim(.Fields("Remarks").Value)) <> UCase(Trim("Passed")) Then
                lv.ListItems.Add .AbsolutePosition, , .Fields("SY").Value, 1, 1
            Else
                lv.ListItems.Add .AbsolutePosition, , .Fields("SY").Value, 3, 3
            End If
            lv.ListItems(.AbsolutePosition).SubItems(1) = .Fields("SEM").Value

```

```

        lv.ListItems(.AbsolutePosition).SubItems(2) = .Fields("Course").Value
        lv.ListItems(.AbsolutePosition).SubItems(3) = .Fields("YR").Value
        lv.ListItems(.AbsolutePosition).SubItems(4) = .Fields("SC").Value
        lv.ListItems(.AbsolutePosition).SubItems(5) = .Fields("Description").Value
        lv.ListItems(.AbsolutePosition).SubItems(6) = .Fields("Units").Value
        lv.ListItems(.AbsolutePosition).SubItems(7) = .Fields("Grade").Value
        lv.ListItems(.AbsolutePosition).SubItems(8) = .Fields("Remarks").Value
        .MoveNext
    Loop
End If
End With

End Function

```

```

Private Sub Form_Load()
    GetCourse
    'set if there is a sy and course
End Sub

```

```

Private Sub GetCourse()
    With DE.rsCourses
        If .State <> 0 Then .Close
        .Open "Select * from courses"
        'clear the combobox
        TxtCourse.Clear
        If .RecordCount > 0 Then
            Do Until .EOF
                TxtCourse.AddItem .Fields(0).Value
                .MoveNext
            Loop
        End If
    End With
End Sub

```

```

Private Function Evaluate(Subject As String, prere As String, msgx As ListView) As Boolean
    Dim Splt
    Dim msg As ListItem
    'check if the subject is taken before
    Dim fnx As ListItem
    Set fnx = LVSubjectsTaken.FindItem(Subject, lvwSubItem)

    If Not fnx Is Nothing Then
        'check if student passed
        Set msg = msgx.ListItems.Add(, , 3, 3)
        If Trim(UCCase(fnx.SubItems(8))) = "PASSED" Then

```

```

'continue
msg.Text = Subject
msg.SubItems(3) = "This has been taken"
Else
'create report here
msg.Text = Subject
msg.SubItems(3) = "Taken before but not passed. Can still take"
End If
Evaluate = True
Exit Function
End If

If Trim(prere) = "" Then 'you can take subject
Set msg = msgx.ListItems.Add(, , 3, 3)
msg.Text = Subject
msg.SubItems(3) = "Can Enroll"
Exit Function
End If
Spl = Split(prere, ", ", , vbTextCompare)
Dim last As Integer
Dim i As Integer
Dim fn As ListItem
For i = LBound(Spl) To UBound(Spl)
Set fn = LVSubjectsTaken.FindItem(Spl(i), lvwSubItem)
If fn Is Nothing Then
Set msg = msgx.ListItems.Add(, , 1, 1)
msg.Text = Subject
msg.SubItems(1) = Spl(i)
msg.SubItems(2) = "Not Taken"
msg.SubItems(3) = "Can't Enroll"

Else
If StrConv(Trim(fn.SubItems(8)), vbProperCase) <> "Passed" Then
Set msg = msgx.ListItems.Add(, , 1, 1)
Else
Set msg = msgx.ListItems.Add(, , 3, 3)
End If
msg.Text = Subject
msg.SubItems(1) = Spl(i)
msg.SubItems(2) = "Taken"
msg.SubItems(3) = fn.SubItems(8)
If fn.SubItems(8) <> "Passed" Then
msg.SubItems(4) = "Not Done"
Else
msg.SubItems(4) = "Ok"
End If

```



```
End If
Next
Evaluate = True
End Function
```

Sample codes to add curriculum

```
Private Sub BtnCancel_Click()
If FrmCurs.isadd = True Then DE.rsCurricula.CancelBatch adAffectAllChapters
SetSelectedCur "Select * From Curriculum where sy = " & FrmCurs.SY & " and sem =
" & FrmCurs.SEM & " and Course= " & FrmCurs.Course & " and yr = " & FrmCurs.yr
& ""
Unload Me
End Sub
```

```
Private Sub BtnOk_Click()
On Error GoTo Erb
setValues
DE.rsCurricula.Update
MsgBox "Record Saved.", vbInformation, "Message"
Unload Me
Exit Sub
Erb:
MsgBox Err.Description, vbCritical, "Error"
DE.rsCurricula.Cancel
'Refresh
'With FrmCurs
' SetSelectedCur "Select * from Curriculum where sy = " & .SY & " sem = " & .SEM
& " and course = " & .Course & " and yr = " & .Yr & ""
'End With
End Sub
```

```
Private Sub Form_Load()
Select Case FrmCurs.isadd
Case True
'add new
If DE.rsCurricula.State = 0 Then DE.rsCurricula.Open
DE.rsCurricula.AddNew
'set values here
With FrmCurs
txtSem.Text = .SEM
txtSY.Text = .SY
txtYr.Text = .yr
txtCourse.Text = .Course
End With
```

Codes used for record adding

```
Private Sub BtnOk_Click()
```

```
On Error GoTo Erb
```

```
'check if it is in the subjects_offered.
```

```
If CheckExistance = False Then Exit Sub
```

```
If CheckPrerequisites = False Then Exit Sub
```

```
setvaluesColRec
```

```
DE.rsSubjectsEnrolled.Update
```

```
MsgBox "Record Saved.", vbInformation, "Message"
```

```
Unload Me
```

```
Exit Sub
```

```
Erb:
```

```
MsgBox Err.Description, vbCritical, "Error"
```

```
DE.rsCurricula.Cancel
```

```
End Sub
```

```
Private Sub BtnSel_Click()
```

```
    FrmSubList.who_Called = 2
```

```
    FrmSubList.Show 1
```

```
End Sub
```

```
Private Sub Form_Load()
```

```
Select Case FrmStudentSubjectList.isadd
```

```
Case True
```

```
    'addnew
```

```
    'setthe values here
```

```
    If DE.rsSubjectsEnrolled.State = 0 Then DE.rsSubjectsEnrolled.Open
```

```
    DE.rsSubjectsEnrolled.AddNew
```

```
    With FrmStudentSubjectList
```

```
        txtfnam.Text = .Fnam
```

```
        txtldnum.Text = .Idnum
```

```
        txtlnam.Text = .Lnam
```

```
        txtmnam.Text = .Mnam
```

```
        TxtSy.Text = .TxtSy.Text
```

```
        TxtSem.Text = .TxtSem.Text
```

```
        Caption = .Lnam & ", " & .Fnam & " " & .Mnam
```

```
        TxtCourse.Text = .Course
```

```
        TxtYr.Text = .yr
```

```
    End With
```

```
Case False
```

```
    'update get values
```

```
    getValuesColRec
```

```
End Select
```

```
End Sub
```

```

Function getValuesColRec()
txtIdnum.Text = DE.rsSubjectsEnrolled.Fields("idnum").Value
txtlnam.Text = DE.rsSubjectsEnrolled.Fields("Lnam").Value
ixtmnam.Text = DE.rsSubjectsEnrolled.Fields("mnam").Value
ixtfnam.Text = DE.rsSubjectsEnrolled.Fields("fnam").Value
TxtSy.Text = DE.rsSubjectsEnrolled.Fields("sy").Value
TxtSem.Text = DE.rsSubjectsEnrolled.Fields("sem").Value
TxtCourse.Text = DE.rsSubjectsEnrolled.Fields("course").Value
TxtYr.Text = DE.rsSubjectsEnrolled.Fields("yr").Value
txtSC.Text = DE.rsSubjectsEnrolled.Fields("sc").Value
txtDescription.Text = DE.rsSubjectsEnrolled.Fields("description").Value
txtUnits.Text = DE.rsSubjectsEnrolled.Fields("units").Value
txtGrade.Text = DE.rsSubjectsEnrolled.Fields("grade").Value
TxtRemarks.Text = DE.rsSubjectsEnrolled.Fields("remarks").Value
End Function

```

```

Function setvaluesColRec()
DE.rsSubjectsEnrolled.Fields("IDNUM").Value = txtIdnum.Text
DE.rsSubjectsEnrolled.Fields("Lnam").Value = txtlnam.Text
DE.rsSubjectsEnrolled.Fields("mnam").Value = ixtmnam.Text
DE.rsSubjectsEnrolled.Fields("fnam").Value = ixtfnam.Text
DE.rsSubjectsEnrolled.Fields("sy").Value = TxtSy.Text
DE.rsSubjectsEnrolled.Fields("sem").Value = TxtSem.Text
DE.rsSubjectsEnrolled.Fields("course").Value = TxtCourse.Text
DE.rsSubjectsEnrolled.Fields("yr").Value = TxtYr.Text
DE.rsSubjectsEnrolled.Fields("sc").Value = txtSC.Text
DE.rsSubjectsEnrolled.Fields("description").Value = txtDescription.Text
If IsNumeric(txtUnits.Text) Then
DE.rsSubjectsEnrolled.Fields("units").Value = txtUnits.Text
Else
MsgBox "Invalid Input.", vbCritical, "Error"
txtUnits.SetFocus
SendKeys "{HOME}+{END}"
End If
If IsNumeric(txtGrade.Text) Then
DE.rsSubjectsEnrolled.Fields("grade").Value = txtGrade.Text
Else
DE.rsSubjectsEnrolled.Fields("grade").Value = 0
End If
DE.rsSubjectsEnrolled.Fields("remarks").Value = TxtRemarks.Text
End Function

```