ABSTRACT

This project is focused on the design of a software system for student recruitment in the Department of Enrollment Management, Texas A&M University – Corpus Christi.

The system provides functionality to search and view a student’s academic, demographic, and admissions data accompanied by an individualized communication with the students who are most likely to attend the University. The software enhances the quality, accuracy, and consistency of communications with prospective students, which ultimately strengthens the student’s relationship with the University. The software helps in initiating and developing personal relationships with potential students. In developing this software, a database of the prospective students was constructed in Oracle9i. The database is a relational database management system (RDBMS) that provides rapid transaction and data sharing in either a single or multi user environment.

The software is designed using Borland JBuilder9. The prospective student database was developed using Oracle9i, since it is a high-performance, cross-platform embeddable database that is used around the world. Combining simple installation, automatic crash recovery, and minimal maintenance, the Oracle database is extremely well suited for database applications.
# TABLE OF CONTENTS

Abstract .......................................................................................................................... ii

Table of Contents .......................................................................................................... iii

List of Figures ................................................................................................................ v

List of Tables ................................................................................................................ vii

1. Introduction and Background .................................................................................. 1

2. Enrollment and Management Systems .................................................................... 8

   2.1 Developing Prospect Student Database ............................................................... 9

       2.1.1 Creating the Prospective Student Database .................................................. 9

   2.2 Designing the Prospective Student Management Software .............................. 11

       2.2.1 Menu Tree .................................................................................................. 12

       2.2.2 Search & Student Data ............................................................................. 13

       2.2.3 Quick Entry of Student .......................................................................... 13

       2.2.4 Quick View of Student Data ................................................................... 14

       2.2.5 Student Demographic Data ..................................................................... 15

       2.2.6 Student Academic Data .......................................................................... 15

       2.2.7 Student Admissions Data ........................................................................ 16

       2.2.8 Majors / Activities Data .......................................................................... 17

       2.2.9 Addresses (Primary / Alternate) Data ...................................................... 17

       2.2.10 High School Information (Primary / Alternate Data) ............................. 18

       2.2.11 College Information (Primary / Alternate Data) .................................... 18

       2.2.12 Student Contact Information Data .......................................................... 20
LIST OF FIGURES

Figure 1.1. Inquiry Pool ................................................................. 5
Figure 2.1. Database Pilot ............................................................... 11
Figure 2.2.1. Menu Tree ................................................................. 12
Figure 2.2.2. Select Student Browser ............................................... 13
Figure 2.2.3. Quick Entry ................................................................. 14
Figure 2.2.4. Quick View of Student Record ..................................... 14
Figure 2.2.5. Student Demographic Data ......................................... 15
Figure 2.2.6. Student Academic Data .............................................. 16
Figure 2.2.7. Student Admissions Data ............................................ 16
Figure 2.2.8. Student Majors / Activities Data ................................. 17
Figure 2.2.9. Student Address Data ................................................ 18
Figure 2.2.10. Student High School Data ......................................... 19
Figure 2.2.11. Student College Data ................................................. 19
Figure 2.2.12. Student Contact Information ...................................... 20
Figure 2.2.13a. Standard File Import .............................................. 21
Figure 2.2.13b. Commercial File Import ......................................... 22
Figure 2.2.14 ACT/SAT Scores ....................................................... 23
Figure 3.1. Entity-Relationship (E-R) Diagram of the System ............... 25
Figure 3.2 Connection Descriptor .................................................... 31
Figure 3.3 Block Diagram of the Java Application Developed ............... 32
Figure 4.1 Borland Debugger .......................................................... 33
Figure 4.2 Search Engine ............................................................... 34
Figure 4.3 SIS version of Academic Data of a Student .............................................35

Figure 4.4 Academic Data (Enrollment & Management version)..........................35
LIST OF TABLES

Table 2.1 ESTUDENT.DBF .............................................................. 10
1. INTRODUCTION AND BACKGROUND

Texas A&M University–Corpus Christi (TAMU-CC) is a rapidly growing University. All the student information is loaded into the Student Information System (SIS) and the mainframe is increasingly loaded with student data. The SIS mainframe is too technical and complex for ordinary users. Therefore, enrollment personnel find it difficult to extract data of prospect students and respond to inquiries made by prospect students to increase the enrollment in the University. For example, when a student requests an application or other information, the requests are entered into the SIS mainframe and are processed every fortnight. The requested information will then be mailed to the students. This sort of communication takes a couple of days to reach the student. A student’s response is much more positive towards a University when a specific communication is being sent to him in a timely manner.

Effective communication with prospects will tremendously escalate the enrollment at a University. To make the most of the limited financial and human resources, successful enrollment management operations must systematically grade and qualify individual student prospects at all stages of the recruiting process – prospect, inquiry, applicant, and confirmed. The goal of enrollment personnel is to direct valuable resources towards aspiring and better qualified students. Getting the right message to the right prospects at the right time is the goal of the Office of Enrollment and Management. Since progressing enrollment is a primary factor in a growing University, effective communication with the prospect students is enormously stressed upon by the personnel of the Office of Enrollment and Management.
There are a number of advanced software systems developed to process prospect student data and to enable effective management of prospect students, like Enrollment Management Action Systems – Recruitment Pro. These systems help universities to target a particular group of students and communicate with them through mail and also offer telecounselling. Using these systems a standard communication plan is set up to share information about the institution with a select group of students [Education Systems 2003].

The first step in communicating with a target group of students using these systems is to establish priorities for the enrollment process for the target student audience. Once the priorities are identified, planning for a communication method that guarantees successful communication with the students is designed. Each student’s communication plan field in the student’s record includes specific contact history as well as planned future communications. This information can be viewed by counselors and other enrollment staff to consistently individualize student communication. In order to effectively, yet economically, communicate with students who want to enroll at the University, students are classified by identifying their intended entry term, market segment, and recruitment cycle stage. A sample cycle of the enrollment process used by other universities is shown below [Noel-Levitz 1997]:

1. Lists of prospects are purchased from a search service such as College Board.
2. These prospects are mailed a personalized letter and a reply card.
3. Reply cards that are returned are entered into the inquiry system where all subsequent communications are initiated.
4. After an inquirer submits an application, his or her information is transferred into the system and is used for all administrative data processing. This record is available to other offices across campus from this point forward.

5. Letters are generated daily for all new applicants.

6. Letters for tracking incomplete applications are sent at regular intervals by the administrative staff. Items needed for the application to be complete are listed in the letter.

7. Acceptance letters are handled manually based on accept conditions. Denial letters are handled manually as well.

8. The Admissions office only uses the file at this point for ad hoc mailings and reports.

As a result of this process, the Office of Enrollment and Management extracted prospect student information from the SIS mainframe, created a prospect student’s information database and also reengineered existing student management software which supports the entire Admissions recruitment and processing cycle. The Admissions module which is a part of the integrated software modules was designed to manage a student’s admission and financial information. The product provides the University with a scalable, efficient and cost-effective solution needed in higher education. This software and database replaced the mainframe services with a database which stores the prospect student information and a PC based Windows environment offering “point - and – click” management.

The communication system provides for easy management of communications with each of the qualified potential students in the University’s inquiry pool.
Acknowledging that unlimited resources are not available to recruit every single prospect, the department needs to think about the University's inquiry pool and how to organize and qualify those potential students. In order to effectively communicate with the students who want to enroll at the University, the personnel at the Office of Enrollment and Management need to classify them by identifying their intended entry term, market segment, and recruitment cycle stage. Once the classification is determined, the next step is to design a communication plan for each of these groups. The following is a sample graphical illustration of an inquiry pool [Figure 1.1]. Explanations of the three identifiers (entry term, recruitment cycle stage, and market segment) follow the illustrations [Noel-Levitz 1997].

![Inquiry Pool Diagram]

**Figure 1.1 Inquiry Pool [Noel-Levitz 1997]**
Entry Term: Entry term refers to the semester, quarter, or session that the potential students would likely enter the University (for example, Summer 2004, Fall 2004, and Spring 2005) [Noel-Levitz 1997].

Recruitment Cycle Stages: The six stages in a recruitment cycle are:

- Prospect (Stage 100) - A student is identified by the University and contacted, but he or she has not responded.

- Inquiry (Stage 200) - The student inquires about the University.

- Applicant (Stage 300) - The student applies to the University.

- Decision (Stage 400) - The student’s application is accepted, denied or waitlisted.

- Confirmed (Stage 500) - The University determines what constitutes confirmation.

- Enrolled (Stage 600) - The student matriculates.

Market segments: A market segment is a set of students who are grouped together by similar characteristics such as geographic location, demographic profile, academic background, interests, or any other grouping attributes. These students are grouped together because a common message will be communicated to them. For example, high academic achievers would need information about the University’s honors program [Noel-Levitz 1997].

Using this software, personnel at the Office of Enrollment and Management are able to communicate effectively with a target group of prospect students by generating personalized e-mail and letters with a simple point and click management. Certain specific queries of students in a particular market segment or in various stages in the
recruitment cycle are resolved with ease using this software. The method of communication helps to initiate and develop personal relationships with potential students.

Every year the University obtains student records from Education Testing Services such as American College Testing (ACT) and Scholastic Aptitude Test (SAT). All the data needs to be finely formatted and inserted into the SIS mainframe. But, extracting student data to set up communications plans with a particular group of students becomes a difficult task.

The new software helps to load the data directly into the database in the correct format and permits easy extraction later to implement plans on a particular group of students. The new software helps in searching for students in the database under their last name, first name, and social security number. It also helps in viewing a particular student's academic, demographic, and admissions data. The demographic data shows the student's current address or previous address, if any. The admissions data shows at which stage the student is in the recruitment cycle. The academic data shows the student's academic records, like the student's scores in tests like ACT or SAT. All these features are implemented through a simple point-and-click in a Windows environment.

The SIS mainframe treats all prospect student information in the same way, but the Enrollment and Management Student Recruitment System groups them by different stages depending on their admissions recruiting cycle. The system also classifies the student into different classifications based on their geographic locations, demographic profile, academic background and interests making it easy for the personnel at the Office of Enrollment and Management to communicate with a particular group of students.
Grouping the students into different categories will help in communicating with a particular group of students comprehensively by maintaining a standard communication plan.

The database for this project was developed in Oracle9i, since it is a relational and cross platform database. The software to access the database was developed in Borland JBuilder9.0, since JAVA is a very powerful database development tool. In this project, the system reads the relevant data every week from the SIS mainframe and manipulates the data into text files so that they can be transferred into the prospect student database. Prospect student records procured by the University from Education Testing Services on a regular basis, are also imported likewise.
2. ENROLLMENT AND MANAGEMENT STUDENT RECRUITMENT SYSTEM

Two major tasks accomplished in completing the Enrollment and Management Student Recruitment System were to construct the prospect student database and to develop the software to manage the database.

The following steps were fulfilled in constructing the prospect student database. Each student in the database is identified by a system generated nine digit StudentId.

- Prospect student database was created using Oracle9i.
- To accommodate the prospect student data, tables were designed within the prospect student database.
- Data was moved from SIS to Oracle9i.

In developing the software to manage the prospect student database, the following applications were developed:

1. Menu Tree.
2. Search of Student Data.
3. Quick Entry.
4. Quick View of Student Data.
5. Demographic Data View of Student Selected.
6. Academic Data View of Student Selected.
7. Admission Data View of Student Selected.
8. Majors/Activities of the Student Selected
9. Addresses (Primary / Alternate) of the Student Selected.
10. High School Information (Primary / Alternate) of the Student Selected.
11. College Information (Primary / Alternate) of the Student Selected.

12. Student Contact Information Data.


14. ACT and SAT Scores of the Student Selected

The uses of these applications are discussed in detail in the sections which follow.

2.1 Developing the Prospect Student Database

2.1.1 Creating the Prospect Student Database

The tables that were created to hold the prospect student database are:

- ESTUDENT: Most descriptive attributes of prospect student information.
- NADDRESS: Student’s addresses primary and alternate.
- EHSCDE1M: Student’s high school attendance information.
- ECOLCD1M: Student’s college attendance information.
- ESCORES: Student’s assessment scores (ACT / SAT).
- EMULTI1M: Student’s multiple code items like majors, activities
- ESTAGE1M: Student’s recruitment stages.
- EST_COMM: Student’s planned Communications.
- ECOMHIST: Student’s communication history.
- EINDVTRK: Student’s Individual Tracking Items.
- EMULTICD: Contains validation codes for majors, activities, and high schools.
- NHS_CDS: Contains validation codes for high schools.
- NCOL_CDS: Contains validation codes for colleges.
- NZIPCODE: Contains all valid postal/zip codes, city, and country information.

- ETERMS: Contains information of current and future entry terms.

- EMAJORS: Contains validation codes for majors and activities.

The following code is used to create the ESTUDENT table in the prospect student database:

```sql
/* Table: ESTUDENT, Owner: SYSDBA */

CREATE TABLE "ESTUDENT"
(
  "STUDENTID" CHAR(9) NOT NULL,
  "SS_NUMBER" CHAR(11) NOT NULL,
  "L_NAME" CHAR(22) NOT NULL,
  "F_NAME" CHAR(20) NOT NULL,
  "M_NAME" CHAR(15) NOT NULL,
  "ISOURCE_CD" CHAR(9) NOT NULL,
  "ENTRY_STAT" CHAR(2) NOT NULL,
  "CITIZEN_CD" CHAR(3) NOT NULL,
  "HS_GRAD_YR" CHAR(4) NOT NULL,
  "ENTRY_TERM" CHAR(5) NOT NULL,
  "STAGE" CHAR(3) NOT NULL,
  "ADMT_CD_DT" TIMESTAMP NOT NULL,
  "BIRTH_DT" TIMESTAMP NOT NULL,
  "VISIT_DT" TIMESTAMP NOT NULL,
  "ACADEM_LEV" CHAR(9) NOT NULL,
  "ADMIT_TYPE" CHAR(9) NOT NULL,
  "COMMUTER" CHAR(9) NOT NULL,
  "CHURCH_CD" CHAR(9) NOT NULL,
  "DENOM_CD" CHAR(9) NOT NULL,
  "EMPLOY_CD" CHAR(9) NOT NULL,
  "EPSS_CD" CHAR(9) NOT NULL,
  "ETHNIC_CD" CHAR(9) NOT NULL,
  "FA_APP" CHAR(9) NOT NULL,
  "FAID" CHAR(9) NOT NULL,
  "GENDER" CHAR(9) NOT NULL,
  "LOADING" CHAR(9) NOT NULL,
  "M_STATUS" CHAR(9) NOT NULL,
  "OREN_SESS" CHAR(9) NOT NULL,
  "PRIORITY" CHAR(9) NOT NULL,
  "PLANMGRP" CHAR(17) NOT NULL
);
```

Table 2.1 ESTUDENT.DBF
Once the tables were created, data was added into them using a small JBuilder9.0 application called **Data Pump**. To check whether the data entered is properly stored in the database, the following command was executed:

```
SQL> select * from student;
```

Borland's Database Pilot was used to browse through the database as shown in Figure 2.1

![Database Pilot](image)

**Figure 2.1 Database Pilot**
2.2 Designing the Prospect Student Management Software

The prospect student management software works in the Windows environment, by accessing the database as the backend. The main menu of the software was designed to be in the form of a menu tree. The menu tree designed helps in the easy navigation through the program. Figure 2.2.1 shows the basic layout of the menu tree.

2.2.1 Menu Tree

The prospect student management software starts with a number of selections in the form of a menu tree and numerous dialogs. The menu tree allows more intuitive navigation of features and student records. The menu tree window is a gateway to many modules of the software depending on the task to be performed or information to be viewed. Each folder on the menu tree contains a set of related tasks. The folder can be opened by double-clicking the folder icon or clicking the plus icon next to the folder icon.

Figure 2.2.1 Menu Tree
2.2.2 Search of Student Data

Listing the Search menu function in the menu tree will make the task of finding a student record quick and easy. Selection of the Search menu brings up the window shown in Figure 2.2.2. To view a student record, Search is selected and the selected student screen appears defaulted to last name search. The student’s last name is entered and ENTER is pressed to search for the student.

![Select Student](image)

**Figure 2.2.2 Select Student Browser**

2.2.3 Quick Entry of Student Data

This module enables the user to enter new student information and add the record into the database. This module’s screen layout is shown in Figure 2.2.3.
2.2.4 Quick View of Student Data

This module allows the user to get a quick view of the data of the selected student from the database. This module’s screen layout is shown in Figure 2.2.4.
2.2.5 Student Demographic Data

This module enables the user to have access to the student’s demographic data. The data consists of important information like the student’s name, SSN, entry status, source code, high school graduation year, ethnic origin and gender. This module’s screen layout is shown in Figure 2.2.5. A user can modify and save changes made to the data.

![Student Demographic Data Screen](image)

Figure 2.2.5 Student Demographic Data

2.2.6 Student Academic Data

This module enables the user to have access to the student’s academic data. The data consists of important information like birthdates, data regarding housing interest and financial aid information. This module’s screen layout is shown in Figure 2.2.6. A user can modify and save changes to the data.
2.2.7 Student Admission Data

This module enables the user to have access to the student’s academic data.

This module’s screen layout is shown in Figure 2.2.7.
2.2.8 Majors / Activities Data

This module gives the user access to the student’s interests in various majors and activities depending on the ranks. A user can add new interests by clicking on the drop down box and selecting the major. This module’s screen layout is shown in Figure 2.2.8.

![Figure 2.2.8 Student Majors/ Activities Data](image)

2.2.9 Addresses (Primary/ Alternate) Data

This module gives the user access to the student’s addresses, both primary and alternate. Entry of the zip code, inserts the relative city, county name and state by default. A user can modify an existing address or add new address to the student’s record. This module’s screen layout is shown in Figure 2.2.9.
2.2.10 High School Information (Primary / Alternate) Data

This module gives the user access to all the high schools attended by the student. A user can search for a high school based upon the high school code, state, city and name of the institution and add the information into the database. This module’s screen layout is shown in Figure 2.2.10.
2.2.11 College Information (Primary / Alternate) Data

This module gives the user access to all the colleges attended by the student. A user can search for a college based upon the college code, state, and name of the institution and add the information to the database. This module’s screen layout is shown in Figure 2.2.11.
2.2.12 Student Contact Information Data

This module allows the user to view and access a student’s contact information. Information such as, the mode of communication (email, letter or phone-call), date on which communication was established, the corresponding response from the student after being contacted, name of the personnel at the Office of Enrollment and Management trying to contact the student are recorded in this module. This module’s screen layout is shown in Figure 2.2.12.

![Figure 2.2.12 Student Contact Information](image)

2.2.13 Import Standard and Commercial Data

This module allows the user to import data into the prospect student database. There are two types of import modules:

- Commercial Imports: Score and Search.

*Standard Imports: Single-file 1 and 2 & Multiple-file*
This option of importing data into the database is used when the data to be imported is from the SIS mainframe. To initiate this type of an import, **Import** from the **Import/Export** menu item is selected, and then **Standard Import** is used to open the dialog as shown in Figure 2.2.13a.

![Standard Import](image)

**Figure 2.2.13a Standard File Import**

**Commercial Imports: Score and Search**

The system can process the following formats:

- ACT Assessment (scores) ACT EOS Plan.
- ACT Search (EOS) ForecastPlus.
- GMASS NRCCUA.
- PSAT/NMSQT Telequalification.
- SAT Scores SAT Search (SSS).
- Phi Theta Kappa Peterson's Regional College.
- Enrollment & Revenue Berkshire Private Colleges.
- Management System (ERMS) & Universities.

This option of importing data into the database is used when the data to be imported is from commercial sources. To initiate this type of an import, Import from the Import/Export menu item, will be selected then Commercial Data to open the dialog as shown in Figure 2.2.13b. For commercial data sources, there are options for assigning a number of default values to each import, including fields such as the Source Code, Stage, Market, and Entry Term.

![Figure 2.2.13b Commercial File Import](image)

Figure 2.2.13b Commercial File Import
2.2.14 ACT/SAT Scores

With the help of this module the user can access a student’s ACT and SAT scores. A user can add new and also update old scores for the students and save it in the database. This module’s screen layout is shown in Figure 2.2.14.

![Scores Summary](image)

**Figure 2.2.14 ACT/SAT Scores**
3. SYSTEM DESIGN

The database server for the Enrollment and Management Student Recruitment
System runs on a RedHat Linux Server and the software application designed runs on a
Windows 98 workstation. Oracle9i is used as the database server and the software
application was developed using Borland JBuilder9.0 [Landy 2002].

The Oracle server runs on a LINUX platform. Oracle recommends a Pentium III
450 with 256 MB of RAM or at least 128 MB. The tables within the database were
created using Oracle tool called sql. The software Borland’s JBuilder9.0 was loaded onto
a Windows 98 workstation. The workstation was used to access the database server. The
software application that was developed runs on the workstation and demonstrates the
features mentioned in section 2.2.

3.1 E-R Diagram

This section describes the design of the prospect student database. The requirements
were first analyzed and an Entity-Relationship (E-R) diagram was designed [Connolly
2001]. A sample E-R Diagram showing the attributes of the table is shown in
Figure 3.1. A student entity has multiple tables holding the student’s information.

- **Student Info** table contains descriptive attributes of prospect students which will
  have only one occurrence.

- **Student Address** contains unlimited number of addresses per student. A student
  may have one primary address and any number of alternative addresses.

- **High School** table contains information about student’s attendance at a particular
  high school.
- **College** table contains information about a student's attendance at a particular college.

- **Scores** table contains information about a student's assessment scores.

- **Multiple Code Items** table contains coded attributes of a student which also includes majors and activities.

- **Stage** table contains a student's enrollment stage.

Each table's primary key is the StudentID which is system generated.

![Diagram of the System](image)

**Figure 3.1 Entity-Relationship (E-R) Diagram of the System**
3.2 Why JBuilder

JBuilder is a good database applications developing tool, since it has a large collection of Java Beans Component Library (JBCL) components. The description of the following sections uses material from Borland JBuilder database developer’s manuals [Borland 1999]. These sections also give an account of the necessary drivers to be set up for the software developed to access the prospect student database.

The database was accessed by JBuilder using JavaSoft’s JDBC API. JBuilder provides a large number of components that are specifically designed to interact with the Oracle server [Borland 1999].

There are many classes defined in the java.sql package which were used by JBuilder to connect to the database. Using these classes, connection was established with the database server. On establishing connection, Structured Query Language (SQL) commands were issued and results from queries were retrieved [Liang 2001].

3.3 Setting up JBuilder with Oracle Server

Steps undertaken to connect to the prospect student database using JBuilder are as follows:

1. Setting up JBuilder to Develop the Software.
2. Adding Oracle Driver to JBuilder.
4. Establishing a Connection with the Prospect Student Database.
5. Connecting to the Prospect Student Database using JDBC.
3.3.1 Setting up JBuilder in Developing the Software

To develop the Enrollment and Management Student Recruitment System using JBuilder, the first step was installation of Oracle JDBC drivers in JBuilder. Like Microsoft’s database connectivity drivers ODBC, Sun developed its own database connectivity drivers called JDBC in conjunction with databases and database tool vendors. JDBC is based upon the X/Open SQL Call Level Interface [Armstrong 1998].

Sun’s JDBC drivers were used in this project instead of Microsoft’s ODBC drivers, since Oracle has JDBC drivers which are endorsed by leading database tool vendors including Oracle, Sybase, Informix, InterBase and DB2. Existing ODBC drivers can be used by using JDBC-ODBC bridge drivers provided by Sun. Using these bridge drivers made the system complicated, since it requires installation of ODBC drivers, registry entries and complex set of code modules. JDBC is an all Java implementation that can be executed directly from a local or centralized remote server. JDBC allows for much simpler maintenance and deployment than ODBC.

3.3.2 Adding Oracle Driver to JBuilder

The following steps were followed when adding Oracle drivers to JBuilder [Borland 1999]:

- Creating a library file containing all the driver classes.
- Setting up the Oracle .config file from the library files to JBuilder’s startup path.
- Add the Oracle library to current project.
The above steps were accomplished by following the detailed steps below:

1. Open JBuilder, select Tools -> Enterprise Setup. Click the **Database Drivers** tab which displays all the `.config` files for all the currently installed database drivers.

2. Click **Add** to add a new database driver, then click **New** to create a new library file for the driver. The library file is used to add the driver to the required libraries list for projects.

3. Oracle was typed and a location was selected for the new file in the **Create New Library** dialog box.

4. Click **Add**, and browse to the location of the driver. Selecting the directory containing the Oracle driver and all its support files, or selecting just the archive file for the driver. JBuilder extracts the information it needs.

5. Click **OK** to close the file browser. This displays the Oracle library at the bottom of the library list and selects it.

6. Click **OK**. JBuilder creates a new Oracle's `.library` file in the JBuilder `/lib` directory with the name `Oracle`. It returns to the Database Drivers page which displays the name of the corresponding Oracle’s `.config` file in the list which was derived from the library file.

7. Close and restart JBuilder so the changes to the database drivers will take effect, and the new driver was put on the JBuilder classpath.

Errors that could occur in adding Oracle driver to JBuilder, are listed in Appendix A.
3.3.3 Setting Oracle Driver to Current Project.

Projects run from within JBuilder use only the classpath defined for that project. This is done from within JBuilder using the following steps [Easttom 2002]:

1. Start JBuilder, close all open projects.
2. Choose Project|Default Project Properties.
3. Select the Required Oracle Libraries tab on the Paths page, and then click the Add button.
4. Select the new Oracle JDBC driver from the library list and click OK.
5. Click OK to close the Default Project Properties dialog box.

The above steps made JBuilder work together with Oracle.

3.3.4 Establishing a Connection with the Prospect Student Database

JBuilder applications connect to remote or local Oracle databases. To connect to a remote prospect student database, the following drivers were needed:

- A JDBC driver for the Oracle server.
- An ODBC-based driver for the server that you use with the JDBC-ODBC bridge software [Calvert 2002].

3.3.5 Connecting to Prospect Student Database

This section discusses how connection was established between JBuilder’s database component and the prospect student database. A database component, which is a JDBC-specific component, manages a JDBC connection, and setting up the properties for this component allowed access to the Oracle data [Bloch 2001]. Steps to add the database component to the current project and connect to Oracle are:
1. Oracle should be running.

2. Open a new application, add a database component or add data connectivity to an existing project and application. JBuilder creates the necessary files and display them in the AppBrowser project pane. The file Frame1.java opens in the content pane. Frame1.java contains the user interface to be designed and the database component for this application.

3. Click the Design tab on Frame1.java at the bottom of content pane.

4. Select the Data Express tab on the component palette, and click the Database component.

5. Click anywhere in the content pane or UI designer to add the Database component to your application.

6. Set the Database Connection Property to specify the JDBC driver, connection URL, user name, and password. The JDBC connection URL is the JDBC method for specifying the location of a JDBC data provider (i.e., Oracle server).

To set the connection property:

a. Make sure the Database object is selected in the content pane. Double-click the connection property in the Inspector to open the connection property editor. Since the data resides on a remote server, IP address of the server is entered here.

b. The following properties were set:

   Driver: oracle.jdbc.driver.OracleDriver

   URL: jdbc:oracle:thin:@enrollat.tamucc.edu:1521:EMAS

   Username: system

   Password: ******
The Figure 3.2.5 illustrates the look of the dialog box which appears.

c. Click the **Test Connection** button to check that the connection properties have been correctly set. The connection attempt results are displayed directly beneath the **Test Connection** button. Click **OK** to exit the dialog and write the connection properties to the source code when the connection is successful [Rozlog 2003].

![Connection Descriptor](image)

**Figure 3.3.5 Connection Descriptor**
The Figure 3.4 illustrates how the software accesses the prospect student database and the layers in the client server Java application to the data source.

Figure 3.4 Block Diagram of the Java Application Developed
4. EVALUATION AND RESULTS

The Enrollment and Management Student Recruitment System that was developed runs on a local machine and is used by the personnel at the Office of Enrollment and Management. It complements the complex SIS mainframe with a simple point-and-click management of the prospect student database. This software enhances the quality, and consistency of communication with the prospect students.

The following steps were performed to evaluate the success of the proposed project [Pressman 1996]:

1. Unit Testing

Objects, classes and methods that were developed in this application were tested to see if they return the desired values, using unit testing and setting up break points in the code with JBuilder Debugger. Sample result of debugging a module that was developed is shown in Figure 4.1.

Figure 4.1 Borland Debugger
2. Integration Testing

Each component of the application was tested when combined with other components and checked for the required result.

3. Interface Evaluation

The user interfaces developed in Chapter 2 were evaluated by the personnel at the Office of Enrollment and Management, to check for the accessibility of the prospect students information. Sample search engine for a student is shown in Figure 4.2

![Select Student](image)

**Figure 4.2 Search Engine**

4. Usability Testing

The usability of the software was tested by the personnel at the Office of Enrollment and Management, in comparison to the existing SIS mainframe.

A sample screen of SIS mainframe showing a prospect student’s academic data is shown in Figure 4.3 and the same module as generated by the software developed
is shown in Figure 4.4.

Figure 4.3 SIS version of Academic Data of a Student

Figure 4.4 Enrollment and Management Student Recruitment-Academic Data
5. FUTURE WORK

No system is complete without wireless, mobile and web access modules which will allow recruiters to take the software system on their laptops or PDA. Since every standalone application developed is becoming a web based application, efforts should be made to make the software application developed in this project web based. Provision of web access, will enable the personnel associated with the Office of Enrollment and Management, to have access to the application if they are on the move. Such modules can be developed in the future. The telecounselling and communications module presently used was developed in Borland Delphi. It can be converted to java and integrated into the Enrollment and Management Student Recruitment System developed in this project.

Web access for students to login and obtain their application information and check their status in the recruitment cycle can also be developed. This would reduce the cost of communication by a great deal. Standard enrollment reports can be customized and integrated into the Enrollment and Management Student Recruitment System.
6. CONCLUSION

The prospect student database which was developed in this project contains data fields that relate to grading and qualifying factors for increasing the student enrollment. This information is easily accessible by users who can update the individual prospect records of each contact, with the help of the Enrollment and Management Student Recruitment System. Data can be retrieved and used to track the progress of the enrollment plans. Data can be imported from commercial sources and standard sources into the database. Using the modules developed in this project, a systematic mode of communication with the target group of students at an appropriate time will help the prospect students in their admission cycle and increase the enrollment at the University. Effective qualification and grading of prospect students in the admission funnel will return huge savings to the University resources that may be wasted upon students who are unlikely to enroll at the University.
BIBLIOGRAPHY AND REFERENCES


Common connection errors and solutions[Li:ang 2001]:

- **Error**: "Unable to locate the Oracle driver".
  
  **Reason**: Oracle Client has not been added as a required library for the project.
  
  **Solution**: Select Project|Properties, and add Oracle Client as a Required Library.

- **Error**: "Driver could not be loaded."
  
  **Reason**: Oracle Client has not been added to the CLASSPATH.
  
  **Solution**: Add the `oracle.jar` file to the JBuilder startup script CLASSPATH