A Windows User Interface to Data Retrieval and Report Generation for a Diabetic Patient Database

Chi-Hwa Kao and Martha Evens
Computer Science Department, Illinois Institute of Technology
Chicago, IL 60616, 312-567-5153, csevens@minna.iit.edu
David A. Trace
UHS/Chicago Medical School, North Chicago, IL 60064
Sant Singh and Subha Yaturu
Diabetes Clinic, VA Medical Center, North Chicago, IL 60064

Abstract

This system is designed to support clinic operations using an MS Windows graphical user interface to provide a menu-driven and interactive operation requiring only mouse clicks to replace the traditional character-based environment for physicians to query from the database and to generate reports. The object-oriented design methodology has made it easy to implement the system and adapt it to changing user needs.

1: Introduction

This system is a prototype making use of an MS Windows graphical user interface to retrieve data and generate patient reports from an ORACLE® database file used in the diabetes clinic of the VA Medical Center, North Chicago, Illinois. This patient database has been built and used for more than 10 years. It includes patient information organized by encounters, observations, diagnoses, laboratory test reports, and drug data. Since it is sometimes still difficult for physicians to learn and memorize lots of keystrokes, commands, and function keys to record every encounter with patients for their medical record information, we have built to provide an easy, simple, and user-friendly interface for physicians to query data and to generate reports.

This system is implemented on a 386-20 DX machine running on DOS 6.0 and Microsoft Windows 3.1. The kernel part uses the ORACLE Pro*C® precompiler to translate embedded structured query language statements so that these statements will be understood by a C++ compiler. The graphical user interface was developed on MS Visual C++ ® and MS Visual Basic® VBX custom control[1] to support the user-friendly interface and menu-driven facility.

2: Design of the query and report generation system

2.1: Embedded Structured Query Language (SQL)

The kernel of this system is an embedded SQL program. We use this engine to access the ORACLE® database and query patient data. This application is more powerful and flexible than applications based on either C or SQL alone [4]. Any SQL statement - data definition, query update, view definition, or index creation - can be embedded in a host language program.

[1] VBX custom controls: An extension to the Microsoft Visual Basic® Toolbox [1].

The embedded SQL statement is distinguished from programming language statements by prefixing it with a special keyword (such as EXEC SQL). In the development stage, we use the ORACLE Pro*C® precompiler to interpret embedded SQL and translate the embedded SQL statements into statements that can be understood by the C++ language compiler.

2.2: Object-oriented programming design in windows environment

This system is based on Microsoft Windows and developed on MS Visual C++. This application development environment (called the application framework [2]) provides several tool facilities that help programmers to build C++ programs. The application framework is an object-oriented interface to Windows that provides the following benefits: [2]

- A significant reduction in the effort of programming an application for Windows.
- Easier use of the Windows Application Programming Interface (API) with C++ than C.
- True Windows API for C++ that effectively uses C++ language features.

These facilities greatly improve the accessibility to each task. By using the object-oriented programming design, we can easily build a window as a class object and handle tasks as objects as well (as shown in Figure 2.1).

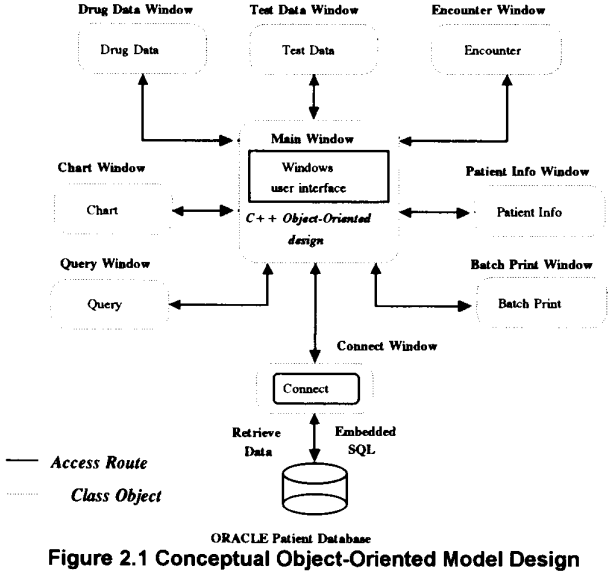

Figure 2.1 Conceptual Object-Oriented Model Design

The application framework supports a foundation class library containing rich sets of C++ classes on which we can build an application for Windows [3]. First, we create a class derived from CwinApp. This class encapsulates the initialization, running, and termination of an application for Windows. We then derive a main frame-window class to build the main window that contains the menu bar, toolbar and scroll window. The application framework will route the command message to the appropriate window function when the user chooses a menu item or clicks the toolbar.

We needed to create some more window objects to handle tasks involving retrieval from the patient database and display that information to user. These class objects are all derived from the CDialog class (containing list box, edit box, scroll bar, button, etc.) to handle various needed events. Each window will carry out a different task (such as displaying chart for patient weight, retrieving the patient social security number for a batch printing task, querying the patient social security number or name to generate report, and displaying the patient encounter information).

2.3: VBX custom control

A VBX custom control window is an extension to the Microsoft Visual Basic Toolbox [1]. It is a dynamic linked library (DLL) and can be called from a C++ program. The custom controls provide many sets of features (such as a gauge to generate a clock, a 3D panel to provide three dimensional look and feel, a graph to display many different types and styles of chart). There are several properties that programmers can set at design time and change further at run time. These features greatly enrich the application program and provide much more visual access. This project associates those custom control objects with C++ object-oriented design to support a user-friendly interface.

3: Example of the query and report generation system

This system starts with a main title window (Figure 3.1). This main window provides two access methods, through the main menu bar or directly from the toolbar.

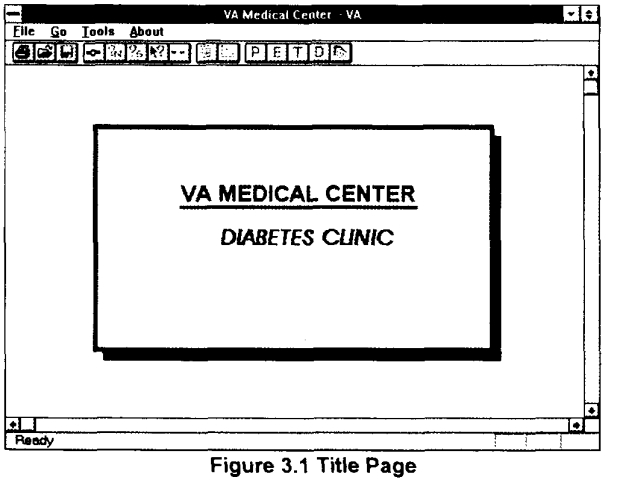

Figure 3.1 Title Page

You can choose "query by social security number". The system will pop up a dialog box to let you input the number (Figure 3.2). You can type exactly 9 digits or 0-8 digits to let the system match the social security number. If the pattern match retrieves more than one person, all the matched social security numbers will appear in the list box. You can always double click the entry in the list box to start the query. Now, the clock icon will give you the current state.

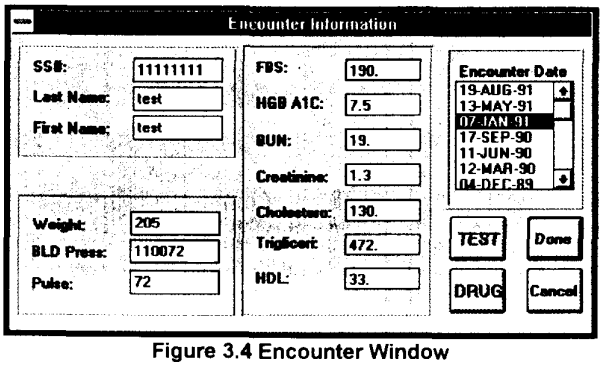

Figure 3.4 Encounter Window

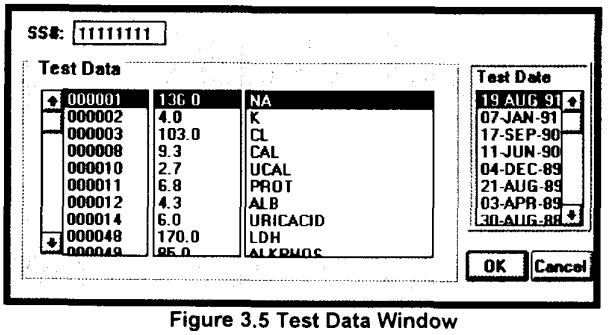

Figure 3.5 Test Data Window

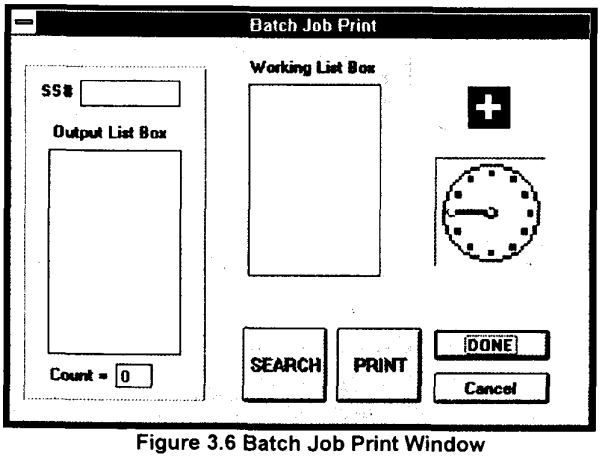

Figure 3.6 Batch Job Print Window

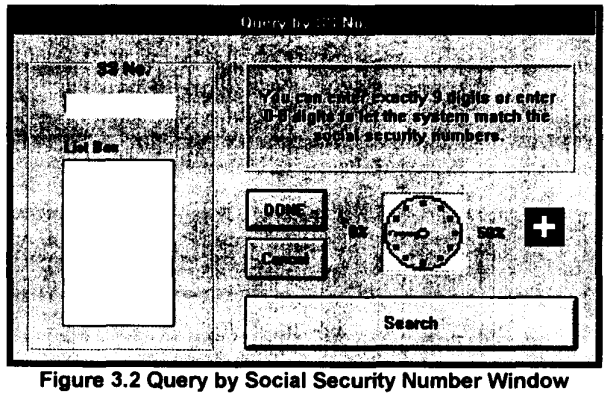

Figure 3.2 Query by Social Security Number Window

You can even choose another method "query by name" (Figure 3.3) to retrieve the patient data. This dialog box supports three methods: query by last name, first name or both. Before clicking the search button, you have to check those items in the query method group box. You can enter any number of characters in the last name or first name box to let the system choose all matched entries. Also, those matched names will appear in the list boxes. Then, you can always double click the last name or first name list box to start retrieving data. The clock icon will show you the current state. Also, you could access this dialog box through the menu bar or through the toolbar.

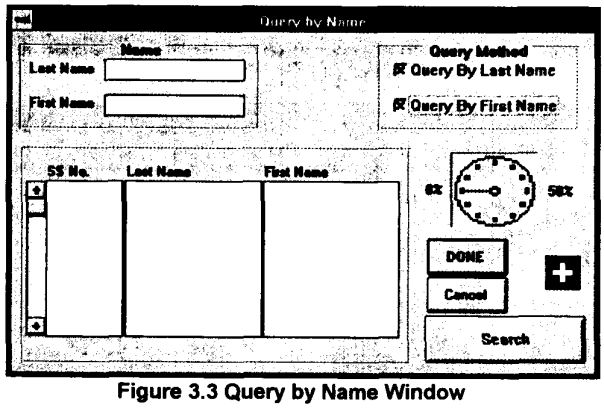

Figure 3.3 Query by Name Window

Now, when finished with the data query process, you can view that data by clicking the toolbar or choosing from menu bar. These dialog boxes will display the patient personal data, diagnoses, observations, encounter data, test data and drug data. You can use the scroll bar to scroll the data that is not visible in the list box. (Figure 3.4 and Figure 3.5 show only two of these dialog boxes.)

A batch printing facility has been requested by the VA Medical Center (Dr. Singh). He wants the computer to carry out a batch job, when he is off duty, Then he could collect these outputs on the next day. You can always enter exactly 9 digits or enter 0-8 digits to let the system match the social security numbers. The working list box will display all matched patient social security numbers. These entries will automatically appear in the output list box, when you double click the entry in the working list box. You could, then, match other patients to add those entries to the output list box. If you are not satisfied with the result that appears in the output list box, you can simply double click on those entries to delete them from the output list box. The count box will show you how many entries have been added to the batch print job, so far. The clock icon also will display the state, after you click the print button to start the print job (Figure 3.6).

4: Conclusion

This system demonstrates a simple, easy, and user-friendly Windows graphical user interface that combines an embedded structured query language, C++ object-oriented programming design and VBX custom controls interface. There will be a trend to develop applications on the Windows graphical user interface, because it provides data interchange, multitasking, device independence, and a user-friendly interface. Those facilities cannot be offered by any traditional character-based environment. Soon, we expect to run Windows on workstations. This capability will give us opportunities to access more data from different machines. As the previous chapter mentioned, this system is a prototype retrieval and report generation system for patient database files. It will grow to accommodate data deletion, insertion, and update. Because it has been developed using an object-oriented design methodology, it will be very easy to expand those functions.

References

1. Microsoft Corporation. 1993. Microsoft Visual Basic Professional Features Book 1. Microsoft Press, Redmond, WA.
2. Microsoft Corporation. 1993. Microsoft Visual C++ Programmer's Guides. Microsoft Press, Redmond, WA.
3. Microsoft Corporation. 1993. Microsoft Visual C++ Reference Volume I: Class Library Reference. Microsoft Press, Redmond, WA.
4. Oracle Corporation. 1990. ORACLE® PRO*C User's Guide, Version 5.1. Redwood Shores, CA.