.NET as a Teaching Tool

John D. Haney
Clinical Professor of CIS
(928) 523-7352
john.haney@nau.edu

John Lovely
Instructor of CIS
(928) 523-7136
john.lovely@nau.edu

College of Business Administration, Northern Arizona University
Flagstaff, AZ 86011-5066

ABSTRACT

In the introductory business programming courses, the BASIC programming language was used for many years followed by Visual Basic more recently. In the intermediate and advanced business programming courses COBOL has been the language used for decades, but it has been replaced by Java more recently. This has come with the benefit of the use of objects, but has also come with the costs of a more awkward Graphical User Interface, and more cumbersome data file management than with COBOL. The use of .NET as a teaching tool solves this problem. The shift from VB to VB.NET in the introductory programming courses is a natural one, and the use of C# in the intermediate programming courses addresses the unwieldy nature of data file management in Java, but retains the benefits of object-orientation. The Graphical User Interface available in Visual Basic is retained in VB.NET, which is available for all languages used within .NET, including C#. The .NET development environment offers a seamless development environment for both introductory and intermediate programming courses.

Keywords: application development, programming, .NET integrated developer environment, introductory classes, intermediate classes

1. INTRODUCTION

The transition from the introductory programming course to the intermediate programming course, in the curriculum of a business oriented computer information systems environment can be challenging. This is especially true where a different language and a different development platform are used, such as when Visual Basic is used in the introductory course and Java is used in the intermediate course. The focus of this paper is to provide a solution to that challenge.

At the core of any business information system are the storage, processing, and retrieval of data. This is the fundamental model of Input-Process-Output. An additional component is data storage, either in flat sequential files or in databases. The type of programs that work with the data falls into two categories: reports (hard copy) and screen displays (soft copy). The pri-

mary language used in business has been COBOL which was specifically designed to manipulate data (Stern & Stern, 2002). The statements and syntax of the language lend themselves easily and intuitively to the processes of data manipulation. The BASIC programming language also has data manipulation statements that are easily understood. Visual Basic (VB) has the same ease of use of sequential data files and the use of relational databases (Bradley & Millspaugh, 1999). So, the use of BASIC followed by VB in the introductory programming course, and COBOL in the intermediate and advanced programming courses was a natural sequence.

Since the programming courses must maintain a business orientation, the continuance of data file usage is essential. The challenge is to provide a content in the programming courses that consists of data file usage, object-orientation, and a GUI, with a smooth transition from the introductory programming course to the intermediate programming course.

2. INTRODUCTORY PROGRAMMING COURSES

BASIC has been the teaching language of choice for introductory programming courses for many years. The core knowledge in the introductory programming courses includes data definitions, assignment of data values, arithmetic operations, logical operations, decision-making, looping, and the usage of arrays. Because BASIC was strictly command line oriented it allowed a focus on the fundamentals of programming.

The introduction of Visual Basic (VB) added a GUI to the BASIC language. The VB language became widely accepted for the development of business information systems, and particularly for User Interfaces. In the introductory programming courses, however, the GUI competed with the fundamentals of programming for time and resulted in a dilution of the fundamental concepts of programming. For the past ten years or so, VB has been the primary language in the introductory programming courses.

With Microsoft's Visual Studio .NET environment a whole new dimension has been added to the BASIC language, that of full object-oriented language. This has transformed the language into a mainline application developer while retaining the features that helped students to understand the underlying principles of accepted programming procedures. In the .NET Integrated Development Environment (IDE) Visual Basic stands shoulder-to-shoulder with C++ and C# (Bradley & Millspaugh, 2003).

Teaching solid programming principles is the primary objective of the beginning programming courses. The first project, which enforces the concepts of correct syntax and naming conventions, is the traditional "Hello World." This program also introduces the concept of the System Development Life Cycle (SDLC). Beginning at the planning phase by identifying what the outcomes are to be and continuing into the analysis of what events will produce which results, the instructor is able to teach conceptual, logical, and physical planning procedures. Providing a structured planning document guides the students through the process. This program can be expanded to include an international business flavor by including radio buttons for selection of languages and displaying a flag icon for the country, a feature often found in E-business websites. Some of the common control objects and their properties that are included in the project are the form, textboxes, labels, option buttons, check boxes, command buttons, and picture boxes.

Subsequent projects include the use of variables, calculations, decision structures, and calling sub-procedures and functions. These projects are oriented on an order form that works within a single form module as in earlier versions of VB. The students can be shown how to step through the code in break mode and watch the flow of data from control to variable and back to control. VB.NET's strict data typing requires the use of data conversion functions and again drives home the concept of object and variable naming conventions. The order form becomes the vehicle for moving the students into the object-oriented world by moving the functions into separate classes. Once objects are introduced all data processing is removed from the main form into class objects. Iterations and arrays are introduced as input and storage media. Multiple form classes can be inherited from student designed base form for a switchboard, invoice,

summary, and help | about purposes. Web forms and ASP.NET can also be introduced at this point. The last projects include storing data in sequential files and retrieving data using the stream writer and stream reader objects. ADO.NET allows the program to store data in a database. The database project is a good candidate for the student to step through the SDLC and develop the capstone project.

VB.NET is a tool that provides all the requirements for teaching the fundamentals of programming and object-orientation, the development of GUIs, and data file manipulation – both sequential data files and relational databases. Use of the form design tools is explored in depth. Decisions, iterations, procedures, functions, data typing, and naming conventions are stressed. Coding syntax and format are reinforced by the IDE, and the debug process is intuitive in the .NET environment (Bradley & Millspaugh, 2003).

3. INTERMEDIATE PROGRAMMING COURSES

In the teaching of business oriented programming, the intermediate programming course is an extension of the introductory programming course. There is an assumption of knowledge that includes the fundamentals of programming: data definitions, assignment of data values, arithmetic operations, logical operations, decision making, looping, and the usage of arrays. Along with the fundamentals of programming there is also an assumption that the introductory programming class includes a certain amount of developing GUIs.

When the programming language of the intermediate programming course was COBOL, the content of the course focused very heavily on data file manipulation. This included flat files with sequential processing, and indexed-sequential files with random processing. Examples of the type of programs that were required are: a file creation program where data is input from the console and then written to a sequential output file; a retrieval type of program where data is read from a sequential input file, and then a report is generated; a data validation program where the input data is written to an output file; the use of a GUI; the use of tables (arrays); a complete update to an indexed-sequential file where records are added, changed, deleted, or queried from the file; and a program where a sub-program is created and called from a main program. As evidenced by the above list of projects, there was a very strong emphasis on data file manipulation, in addition to the use of GUIs and arrays.

COBOL originally was command-line oriented, so the user-interfaces were not graphical (Stern & Stern, 2000). With the advent of VB many user-interfaces were developed in VB that dovetailed with legacy systems written in COBOL. Therefore, the use of VB in the introductory programming courses and COBOL in the remaining programming courses was an easy decision to make. Eventually, a Graphical User-Interface (GUI) was incorporated into COBOL, such as MicroFocus COBOL (Lorents, 2000). This added the dimension of a GUI into the intermediate and advanced courses. What remained at the core of the intermediate and advanced courses was a strong focus on data manipulation, either in sequential files or with databases.

When COBOL was replaced by Java in the intermediate programming courses the content of the courses changed dramatically (Course Equivalency Guide, 2003). There was a strong focus on object-orientation along with the use of GUIs, and some data file usage. Depending on the instructor there was a trade off between the GUI and data file usage. Examples of the type of programs that the students are required to write include: a review program that provides an overview of object components - classes, data members, and data methods, with a focus on parameter passing; arithmetic processing with a focus on objects; the use of supplier classes; the use of inheritance, polymorphism, and arrays; sequential data file usage; and a program that interacts with a database for random access processing. The above list of projects shows a strong emphasis on object-orientation, the use of arrays, and some data file manipulation.

The use of Java, in the intermediate programming course, brought both positive and negative changes. The first positive change

was object-orientation because Java is a totally object-oriented language (Staggered, 1999). The popularity of Java in the workplace makes the language a natural inclusion in the curriculum. However, Java also came with some negative effects. The primary negative effect was the loss of focus on data manipulation in the intermediate programming courses. Data manipulation is possible in Java, but the process is not as straightforward as in COBOL. The added focus of objects in Java also diminished the time availability for data manipulation. Another negative effect of using Java is the GUI. The Advanced Windowing Toolkit (AWT) and Swing are not as intuitive as the development environment in VB (Bradley & Millspaugh, 2000) or even the Dialog system in MicroFocus COBOL (Lorents, 2000).

The transition of the programming language in intermediate programming courses from Java to C# combines both the content of using COBOL and Java. The intent is to bring back a heavy emphasis on data file manipulation and the inclusion of GUIs, without the loss of object-orientation. Examples of the type of programs that students are required to write, in addition to the review of fundamentals and object-orientation using Java, are: the development of GUIs; writing to a sequential output data file; the use of fixed-length and comma delimited sequential data files; the retrieval of data from a relational database; the use of arrays; and a full update to a database table by adding rows, changing or deleting rows, and querying a single row. As evidenced by the list above, there is a strong emphasis on data file manipulation, in addition to a substantial use of GUIs, the use of arrays, and retention of object-orientation that combines the strength of both COBOL and Java.

4. SUMMARY AND CONCLUSIONS

The .NET platform provides a teaching tool that offers a seamless transition from the introductory programming courses to the intermediate programming courses. The foundational concepts of programming, object-orientation, Graphical User Interfaces, and data file manipulation laid down in the beginning course can be built upon in the intermediate course in a more systematic and logical fashion by using the same development environment. The interactive development environment is also the same for creating Windows applications or web applications. The .NET platform allows for the inclusion of several programming languages, including Visual Basic and C#. Therefore, as the students go from an introductive programming course to an intermediate programming course the learning curve is greatly reduced from the current situation of going from Visual Basic to Java.

Since BASIC followed by Visual Basic has been the programming language of the introductory programming courses it was a natural decision to replace Visual Basic with Visual Basic .NET. The decision concerning which programming language to use in the intermediate programming courses has been more problematic. For many years the decision was an easy one because COBOL was the primary language for developing business information systems. With the arrival of Java, and the development of web-based information systems Java replaced COBOL in the intermediate programming courses. This decision brought both positive and negative consequences. The main benefit was the addition of object-orientation to the curriculum. The costs were a diminished emphasis on data file manipulation and a more awkward Graphical User-Interface. By changing from Java to C#, the emphasis on objects has been retained, but the re-emphasis on data file manipulation has been greatly enhanced, along with a much greater weight given to Graphical User Interfaces.

By using the .NET platform as a teaching tool in both courses the challenge has been met. The development environment remains the same, and the development of Graphical User Interfaces is consistent in both courses. In addition the foundational concepts of object-orientation can be presented in the introductory course and fully developed in the intermediate programming course.

5. REFERENCES

Bradley, Julia Case and Anita C. Millspaugh, 1999, Programming in Visual Basic 6.0. San Francisco, CA: McGraw-Hill-Irwin.

Bradley, Julia Case and Anita C. Millspaugh, 2003, Programming in Visual Basic .NET. San Francisco, CA: McGraw-Hill-Irwin.

Bradley, Julia Case and Millspaugh, Anita C. (2003). Advanced Programming using Visual Basic .NET. San Francisco, CA: McGraw-Hill-Irwin.

"Course Equivalency Guide" (2003), Online: http://az.transfer.org/cas/students/transfer_guides.htm.

Deitel, H.M., P.J. Deitel, J.A ListField, T.R. Nieto, C.H. Yaeger, and M. Zlatkina, 2003, C# A Programmer's Introduction. Upper Saddle River, N.J.: Prentice Hall.

Farrell, Joyce, 2002, Visual C# .NET. Canada: Course Technology.

Lorents, Alden C., 2000, Elements of Dialog System Using Micro Focus Net Express. Orinda, CA. Object-Z Publishing.

Sharp, John & Jon Jagger, 2002, Visual C# .NET. Redmond, WA.: Microsoft.

Staggered, Andrew C. Jr., 1999, Java for Computer Information Systems. Upper Saddle River, N.J.: Prentice Hall.

Stern, Nancy and Robert A. Stern, 2000, Structured COBOL Programming. New York, N.Y.: Wiley.