Integrating High Level Programming Languages For Teaching Business Application Development

Allen B. Zilbert[1]
Mathematics, Computer Science And Computer Information Systems Department
Molloy College
Rockville Centre, NY 11571-5002

Abstract

Over the last thirty years, the methodology for teaching computer programming has changed. Previously implemented procedures teaching one computer programming language has moved towards the educating of students with a variety of computer languages. During this same period of time, we also have seen the business environment move towards the usage of a variety of computer programming languages. However, the graduating student population is not achieving the same knowledge when one computer language is implemented compared to using several application development tools. The focus of this paper will be to compare the two methodologies that have been used as the instructional tool for educating students in the discipline of computer programming. Also, this paper will recommend an alternative teaching procedure in which computer application development can be accomplished in the classroom environment.

Keywords: BASIC, COBOL, C++, Computer Programming Curriculum, Programming Language Integration.

1. INTRODUCTION

Today, within the business environment, there is a multitude of programming languages that are implemented for software development. Currently, organizations are using such popular languages as COBOL, RPG, JAVA, C, C++, Visual BASIC, APL, FORTRAN, MUMPS, ADA, PASCAL as well as numerous other high level (third generation) programming languages. According to the Master List of Languages for the Dictionary (Master List, 2001) and the Language List (Ulogov, 2001), there are approximately over 2,200 computer-programming languages in existence today. This list is constituted of different generation programming languages, which ranges from first generation through fifth generation programming languages. Also, the composition of the list is of different versions of the same language such as FORTRAN I through FORTRAN 90. Finally, this listing provides different dialects of the same language. As an example, the BASIC programming language that was formulated and conceived by John Kemney and Thomas Kurtz in 1964 at Dartmouth College has evolved into numerous offspring's from its predecessor. Table 1 provides an illustration of the current listing of the BASIC programming language. Still, there are other companies that are maintaining applications that were designed in Assembly language (second generation). The problem that business computer information systems programs are faced with is what computer programming language to choose for educating students who either intend to derive their livelihood from application development (i.e. become computer programmers) and for those students that are information systems majors (non programming track) that are to be cognizant of what computer programming has to offer and how it has to be interfaced within the business environment.

2. COMPUTER PROGRAMMING MODEL

As one is well aware, it is physically impossible as well as inappropriate to teach computer information system majors, just computer programming. Computer curriculums must include a diversity of computer courses such as systems analysis and design, database management systems, telecommunication and networks

[1] azilbert@molloy.edu

that will provide a well rounded computer education for business majors. The curriculums implemented by most schools are very similar in description. Over the years, professional organizations such as AITP (Association of Information Technology Professionals), ACM (Association For Computing Machinery), and AIS (Association for Information Systems) have formed an alliance in order to provide the necessary core for the computer information systems discipline (Model Curriculum, 1997). These three organizations are cognizant of the needs of the business community. As a result, their intention is to provide schools with the appropriate model curriculum that will support the growing requirements of the business community. Presently, undergraduate degree programs in information systems are able to follow the IS 97 model curriculum.

Most colleges and universities today require computer information systems majors to take either two or three programming courses. As indicated in the IS 97 mode curriculum, one programming course is advocated with respect to programming proficiency. However, programming is stressed through several courses within their curriculum structure. Also, it should be noted that this prototype does not provide a recommendation in terms of which computer language should be implemented in terms of the instruction. The majority of schools of business with a computer information systems program offer application development courses in Visual BASIC, C++, and COBOL. An illustration of the courses that emphasize computer programming, which are found in the IS 97 model curriculum for either the instruction of computer programming or as the prerequisite for a course is shown in Table 2. Within the eleven courses that are described by this paradigm, four out of the eleven emphasize the implementation of computer programming.

Current Dialects Of The BASIC Programming Language For The Microcomputer				
Applesoft BASIC	EBASIC	L-BASIC	RBASIC	Utah BASIC
bs	Future BASIC	MBASIC	RealBASIC	Visual BASIC
CBASIC	GW-BASIC	Omikron BASIC	Tiny BASIC	Waterloo BASIC
Chipmunk Basic	Info BASIC	Power BASIC (Turbo BASIC)	True BASIC	XBASIC
Dartmouth BASIC	KBASIC	QBASIC	UBASIC	ZBASIC
Data/BASIC (Pick BASIC)	Liberty BASIC	Quick BASIC	UNBASIC	

Table 1 – Versions Of The BASIC Programming Language

The Usage Of Computer Programming Within The IS 97 Curriculum Model		
Course Number	**Course Title**	**Course Type**
IS'97.5	Programming, Data, File And Object Structures	Instructional Programming
IS97.8	Physical Design And Implementation With DBMS	Programming Background Required
IS97.9	Physical Design And Implementation With A Programming Environment	Programming Background Required
IS97.10	Project Management And Practice	Programming Background Required

Table 2 – Programming In IS 97 Model Curriculum

Twenty years ago, students would have to take either a beginning course in programming followed by an advanced course in programming. The schools that would mandate three courses normally had a beginning, intermediate, and an advanced course in programming. Usually, one programming language was selected as the teaching tool to provide the learning experience for the students. The trend with respect to using one programming language has changed over the last ten years. Now, educational institutions are implementing between two and three different languages with respect to the programming courses.

3. ADVANTAGES OF TEACHING MORE THAN ONE COMPUTER LANGUAGE

There are several advantages in teaching more than one programming language to students who intend to make their careers as computer programmers. The most important advantage that one must address is diversity. Most organizations do not program in just one computer language. Granted, companies will have a primary environment for developing their software. However, this principal language may not easily support certain application creation. Organizations as a result will turn to other computer languages to subsidize their development requirements. In order to enhance their marketability to companies, students having knowledge of more than one programming language will be extremely desirable to these companies. Also, not all industries will create software implementing the same principal language. Different areas of specialization tend to dominate one computer language over another. For example, hospitals have the majority of their application development prepared in either MUMPS or COBOL. In the early 1960's, the MUMPS language was explicitly designed for the medical industry. The name MUMPS is an indicator that it was produced for the medical platform. MUMPS comes from the acronym Massachusetts general hospital Utility Multi-Programming System. However, financial institutions have their software development written in COBOL, C, and C++. In business, the legacy language COBOL (COmmon Business Oriented Language) is even now striving as an instrumental software development tool for over forty years. But, organizations in business are not sitting quietly. Many firms are supplementing their program creation by either implementing JAVA or J++. Other organizations are contemplating whether they should commence program development in a new language from Microsoft called C# (Truppin, 2000). Still today, actuarial programs are being designed in the scientific language APL.

Another benefit for instructing students with more than one language is the opportunity to use features that are only prevalent within that language. One of the major benefits for using C++ for application instruction is the ability to use a programming option known as overloading (Zak, 2001). This characteristic of the C++ language allows for the development of procedures and functions with non-unique names. As long as the parameters are different for these operations, the names can be the same. An example of overloading procedures in C++ for the purpose of counting the number of items of different data types within a list is shown in Figure 1. Unlike other application development tools, the COBOL programming language has a built in operation that permits for the sorting of data (Stern, 2000). Normally, programmers would have to write their own algorithms in other programming languages in order to accomplish the sort process. However, invoking a few simple commands immediately generates the function that will arrange the data into any desired order. Figure 2 illustrates the command implemented by COBOL to collate data. Designing a front-end application program interface can be a tedious as well as time-consuming procedure for a computer programmer. By implementing Visual BASIC, for the purpose of developing the GUI (Graphical User Interface), the screen input design can be expedited. Visual BASIC is enhanced with the tools to simplify screen design (Zak, 1999). Also, it provides the integration of the video input and output with respect to event trapping in order to process the users' data.

Visual BASIC, C++, and COBOL also have other attributes that are unique to their languages respectively. However, the purpose of this article is not to illustrate the features that one can benefit from learning each of these languages. The objective is to provide an understanding of the benefit for learning more than one computer programming language during the students' study of application development.

An additional rationale for educating students with more than one computer language is that some vernaculars will be more efficient than other dialects while performing the same exact operation. As an example, file operations are available within all computer languages. However, COBOL is one of the most efficient languages that can be implemented for the purpose of input and output. File handling is prevalent within C++, but this functionality is extremely weak for this purpose. On the other hand, it is more desirable to perform calculations in C++ rather than it is to do the same mathematical operations in COBOL. C++ provides higher precision with respect to financial computations.

4. ADVANTAGES OF TEACHING ONE COMPUTER LANGUAGE

Educating students is simplified when one programming language is implemented within the curriculum. Using one language facilitates the instruction of a significant amount of material. Advanced programming topics can be discussed in detail once students are provided with the foundation material. Students' proficiency in

application designed is strengthened when a sufficient amount of time is spent using one computer

Overloading In C++

```
void count (int [ ]);
void count(short int [ ]);
void count (long int [ ]);
void count(float [ ]);
void count(double [ ]);
void count(char [ ][ ]);
```

Figure 1 – Declarations Implemented In An Overloading Operation In C++

Sorting In COBOL

```
SORT employee-file
        ON ASCENDING KEY
                sort-employee-last,
                sort-employee-first,
                sort-employee-mi
        COLLATING SEQUENCE IS sort-order
        INPUT PROCEDURE IS unsorted-rtn
        OUTPUT PROCEDURE IS sorted-rtn.
```

Figure 2 – COBOL Syntax For Sorting File Data

programming language. Highly sophisticated and organized programming projects can be assigned, which can span within a two-semester course sequence (Bend, 1990). Table 3 illustrates the curriculum that normally would be covered in a two-semester programming course curriculum.

When multiple languages are taught in a programming curriculum, the topics found in Table 3 are discussed in a one-semester time period. However, the majority of students do not grasp most of the necessary programming concepts. There is a higher student satisfaction as well as ease in the understanding of programming as a result of multiple semesters of the same language. When an enormous amount of information is doled out during a one-semester course, the majority of students taking the course tend to result in disliking programming. On the contrary, when more than one-semester is implemented for teaching computer programming, there is a higher fondness of application tool design.

5. AMALGAMATING THE PROGRAMMING TOOL

Currently, when students are taught more than one language, they are reintroduced to programming logic that was previously covered in a different programming course. As has been discovered in other educational findings, the reinforcement of teaching the same material does benefit student learning (Sutton, 1998). However, the question that needs to be addressed is how much reinforcement should students be receiving. Many educational institutions will permit students to take several courses in programming. In fact, these students are being introduced to a variety of programming tools as an introductory perspective. However, these same students are not being given the opportunity to address the advanced material mandated in order to be able to get a job. When advanced topics are encountered in the multi-language programs they are either covered expeditiously or too minutely. In either case, students are not provided the appropriate education that they deserve.

Computer Programming Curriculum	
Topic	**Level**
Data Names and Constants	Introduction
Arithmetic Operations	Introduction
Conditional Statements	Introduction
Report Formatting	Introduction
Classes	Introduction or Advanced
Loops	Introduction or Advanced
File Handling	Introduction or Advanced
Arrays	Advanced
Pointers	Advanced
Character Manipulation	Advanced
Sorting	Advanced
Searching	Advanced

Table 3 – Introductory And Advanced Programming Topics

Selecting either just one computer programming language or several programming languages for instructional purposes is not the answer. Both methodologies do have positive positions. However, both options also have negative arguments. What is mandated is an integration of the positive attributes of the aforementioned procedures. It has been proven that it is easier for children to learn to speak more than one language when they are at a very young age. Why not use the same technique, and teach more than one programming language within the same course.

In terms of feasibility, it would not be possible to teach all of the components of several languages within one, two, or even three course. However, it would make logical sense to provide the most important components of each language in a new application development curriculum. By implementing a three-semester sequence of computer programming, it would be possible to incorporate the subject matter that is relevant from the languages C++, Visual BASIC, and COBOL.

Students would be introduced to the components that are applicable to that aspect of the computer program. Also, students would learn how to interface different computer languages. Currently in business, the incorporation of diverse computer programming languages for the function of software development has become a conventional methodology for writing an application (Rippin, 2001). However, computer-programming curriculums do not cover the interfacing of computer applications using diverse programming languages. There would be a primary benefit for students to be educated in how computer programs can have their data shared by passing parameters between different computer languages. For example, Visual BASIC could be implemented as the front-end of the application. C++ could be taught in terms of the processing components of the program. And, COBOL could be used as the back-end of the application.

6. CONCLUSION

There are two objectives that must be focused on within the present computer-programming curriculum. The first is the educating of the students, and the second is fulfilling the requirements for the business environment. These two goals can be culminated simultaneously by revising the structure of the current curriculum. By joining the necessary components of each of the most widely implemented computer-programming languages (COBOL, Visual BASIC and C++), students will be instructed appropriately. This in turn will result for the business sector to acquire a pool of graduating college students who will possess the knowledge mandated for their entry-level job positions. In either case, two problems will be solved from the modifications that are mandated within the computer-programming curriculum.

7. REFERENCES

Application Extension Product Capability Matrix, 2001, http://www.merant.com/products/microfocus/product_focus/techwhite.asp

Bend, Robert., 1990, Basic: An Introduction to Computer Programming, Brooks/Cole Publishing, California.

Defining Computing Curricula for the Modern Age Computer, 2001, http://www.merant.com/products/microfocus/.

Forsyth, Richard., 1978, The Basic Idea, Chapman and Hall Publishing, New York.

Mashaw, Bijan., 1985, Basic, Mayfield Publishing Company, London.

Master List of Languages for the Dictionary, 2001, http://cgibin.erols.com/ziring/cgi-bin/cep/cep.pl?_get=epl_masterlist.phtml.

Model Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems, 1997, www.aitp.org.

Rippin, Wayne., 2001 Developing Mixed Visual BASIC/ COBOL Applications, http://www.merant.com/products/microfocus/product_focus/techwhite.asp

Stern, Nancy and Stern, Robert. 2000. Structure COBOL Programming, John Wiley And Sons, New York.

Sutton, Richard S. and Andrew G. Barto., 1998, Reinforcement Learning: An Introduction, MIT Press, Massachusetts.

Truppin, Joshua., 2000, Sharp New Language: C# Offers the Power Of C++ And Simplicity Of Visual BASIC, MSDN Magazine, www.microsoft.com/msdmag/issues/0400/vbnexgen/print.asp

Truppin, Joshua., 2000, The Future Of Visual BASIC: Web Forms, Web Services, and Language Enhancements Slated For Next Generation, MSDN Magazine, www.microsoft.com/msdmag/issues/0400/vbnexgen/print.asp

Ulogov, Vladimir I., 2001, Language List, oop.rosweb.ru/Other/

Zak, Diane. 2001.An Introduction To Programming With C++, Course Technology, Massachusetts

Zak, Diane. 1999. Programming With Microsoft Visual BASIC 6.0, Course Technology, Massachusetts.