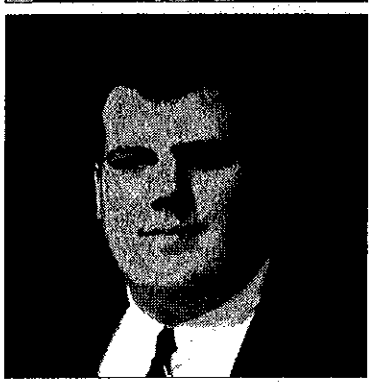
Journal of Information Systems Education
Summer 1995

WINNER
Best Paper
TRACK 3
Worldwide Conference on Information Systems Education
EDSIG
1995

Restructuring Programming Instruction in the Computer Information Systems Curriculum: One Department's Approach



ABSTRACT: The rationale for and details of one Computer Information Systems (CIS) department's plans for a drastic restructuring of the CIS curriculum are presented. The proposed approach is compared with current and developing model curricula for both computer science (CS) and computer information systems programs. The new curriculum's approach to information systems construction is characterized by delivering training in the use of fourth generation development tools, the assembly of software components, event-driven programming and client/server practices. The development tools, the programming environment and the client interface are all equipped with a graphical user interface (GUI).

KEYWORDS: *Programming, Curriculum, C++, Visual Basic, Object-Oriented GUI*

John K. Gotwals
Carlin R. Smith Jr.
Department of Computer Technology
Purdue University
West Lafayette, IN 47907-1421
email:
jkgotwal@mailpo.cpt.purdue.edu
admin@mailpo.cpt.purdue.edu

INTRODUCTION

Many CIS and MIS departments are producing graduates with programming skills which include batch oriented COBOL, VSAM file processing, interactive on-line COBOL using CICS, and host based processing. Their graduates learned these skills while working in a "command line" program development environment, and the programming tools and the applications they produced executed in a character based, as opposed to graphical, mode. While seeking employment, the graduate (or prospective graduate) is shocked to learn that corporate restructuring and "downsizing" has produced large numbers of job seekers who have these same skills. The graduate also discovers that prospective employers are scanning their resumes looking for generic keywords such as GUI front-ends, object-oriented, client/server, RAD (Rapid Application Development)prototyping and specific keywords such as Visual Basic, Powerbuilder, C++, Oracle, Windows and UNIX. James Martin has stated that academia is teaching obsolete languages and methodologies and is ignoring current technologies (cited in Hensel, 1994).

It is the purpose of this paper to point out the shortcomings in the programming portion of CIS curriculums which are similar to ours, to list the skill set that would be more useful to students, and to describe a sequence of programming courses which will help CIS students acquire this skill set.

SHORTCOMINGS IN THE CURRENT CURRICULUM

COBOL is the First Programming Language

There appears to be some disagreement among MIS educators as to the future of COBOL. One information systems educator states that "it is apparent that COBOL will remain the overwhelming language of choice for business applications during the foreseeable future", and then goes on to urge the revision of introductory business programming courses from batch oriented COBOL programming instruction to "on-line" CICS COBOL programming instruction (Brumbaugh, 1994). An alternate view is expressed by Lauer and Graf (1994) who received 119 usable survey returns from a national group of MIS managers and concluded that COBOL usage in new applications will drop from its current value of 44 percent to a value of 28 percent in five years.

In our current curriculum a student first learns about programming by enrolling in a programming core course which teaches structured programming. In this course a student learns how to design programs by using the restricted control structures to either write pseudo code or draw flow charts which describe a solution to the assigned problem. After the design is finished, the student is taught how to use the programming language COBOL to implement the design and then test and debug the program. The programming environment is entirely PC hosted and uses the Micro Focus COBOL workbench.

In the following semester, the student takes a second programming core course which exposes the student to VSAM file processing and the MVS operating system. The course devotes a limited amount of instruction to software concepts such as module cohesion and coupling, and the source program is almost always contained in one file. In both courses, the student works in a character based environment and produces programs which either run in a batch mode or have a very rudimentary form of character based interactive input/output.

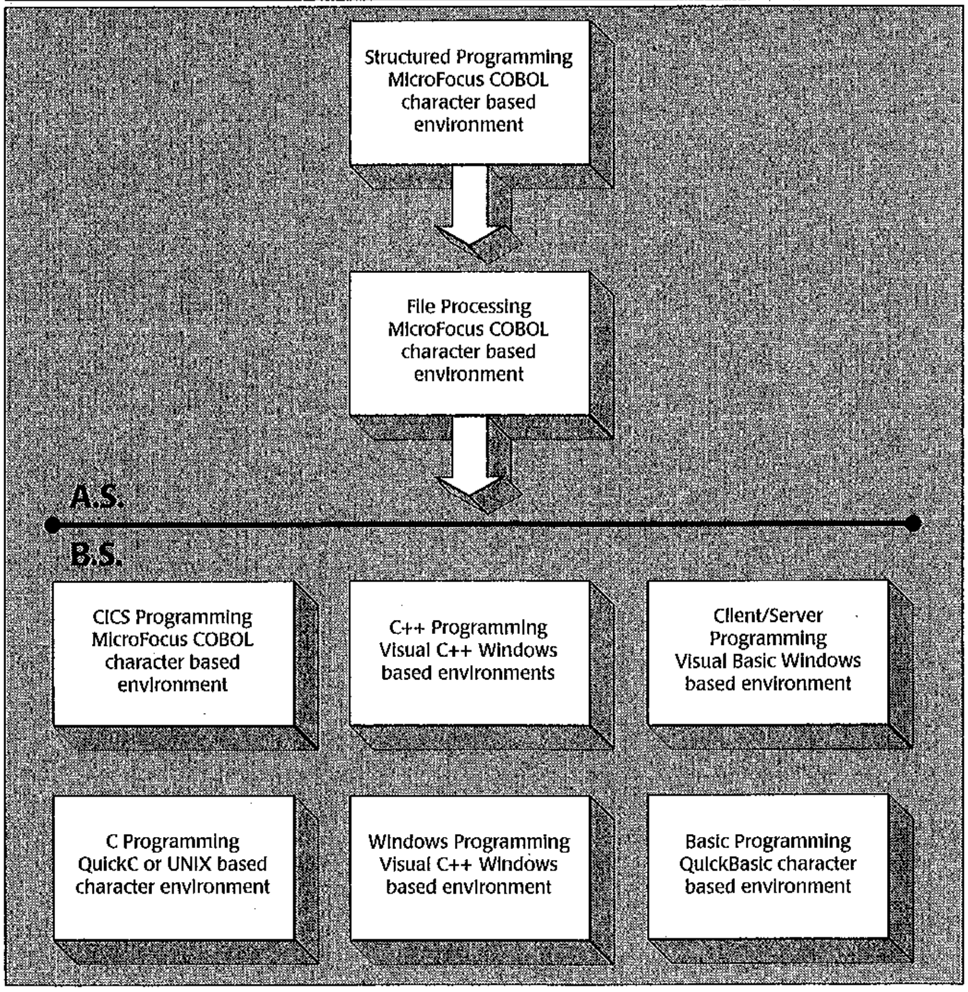
Because of the popularity of the programming language C, many of these students enroll in an elective C programming course. Since this course has a prerequisite of one programming course in any language, the instructor assumes that the students already understand the concepts of programming. This class draws students from the entire university with an assortment of language backgrounds such as BASIC, COBOL, FORTRAN and Pascal. Since most people learn new things by relating them, if possible, to previous knowledge, almost all instructors of this course have observed that many of the students who have only had experience with COBOL have a difficult time mastering C. We believe this occurs because COBOL does not introduce the student to the following concepts:

- Scope and extent (lifetime) of variables - In COBOL, all variables have global scope and static extent.
- Functions and function parameters - Most students who learn COBOL are not taught about the CALL statement and hence the concepts of functions and parameters are foreign.
- Pointers - Many students have some problems with understanding and using pointers, but students with a Pascal or PL/I background have already had exposure to this very important topic.

Important Topics are Taught as Electives

Because of the rapidity of change in the CIS field, there is a propensity for an undesirable time lag to develop between curriculum content and current practice in industry. Indeed, we found that our curriculum required students to take courses which had obsolete course content while the courses which had more relevant material were offered as electives! Topics

FIGURE 1


Structured Programming MicroFocus COBOL character based environment
File Processing MicroFocus COBOL character based environment
A.S.
B.S.
CICS Programming MicroFocus COBOL character based environment
C++ Programming Visual C++ Windows based environments
Client/Server Programming Visual Basic Windows based environment
C Programming QuickC or UNIX based character environment
Windows Programming Visual C++ Windows based environment
Basic Programming QuickBasic character based environment

which are taught in electives but which should be moved into the mainstream of the curriculum include:

- GUI database front-end development with high-level tools such as Visual Basic or Powerbuilder
- GUI Design
- Event-driven programming
- Assembling systems from software components
- Client/server programming
- Object-oriented programming using C++
- Windows Programming using Class Libraries

Lack of Large or Complex Development Projects

Students do not get experience in the development of a large projects, and for this reason they do not come into contact with project management, and version control software, and test data generators. Furthermore, they are not exposed to instruction in formalized testing procedures. Many students can formalize solutions to problems in their mind because the projects lack sufficient size and complexity to challenge all but the weakest students. Students complete projects that are large in physical size, but the complexity is low.

No Critical Path Exists for Programming

Students are required to take two courses in COBOL at the freshman level. All electives can be taken in any order that the students wishes. This forces the electives to naturally duplicate some coverage of material between classes. Since electives are taught using Windows based environments, event-driven programming principles have to be covered repetitively in each course. The lack of a critical path reduces the amount of incremental learning that can be achieved in a single semester.

CURRENT COURSE SEQUENCE

Figure 1 depicts our current programming curriculum. The first two programming courses are required for all students and are normally taken during the freshman year. All other courses are electives, and no other programming course is required to complete a baccalaureate.

PROPOSED COURSE SEQUENCE

Figure 2 depicts a programming track as part of an information systems curriculum. Students who choose to enroll in the track would be required to complete five sequential courses in programming. Students not electing the programming track would be required to take a COBOL elective as their third programming course.

Introduction to Programming

The students' first experience with programming comes during their third semester. Because of this placement, students will have already completed courses in Windows, database, and information literacy. Visual Basic Professional for Windows will be used as the language and development environment. Other educators have made the observation that when suitable textbooks are published, Visual Basic will be an excellent programming language choice for both the first and second courses in programming (Newman and Boman, 1994). Choosing a first language to replace COBOL is an unnerving task, since COBOL has enjoyed virtual dominance in information systems for over twenty years. It is our belief that there may never be another language that will attain the dominance of COBOL. Visual Basic was chosen for the following reasons:

- Excellent Integrated Development Environment
- Easy language to grasp
- General purpose
- Immediate feedback by eliminating long compile and build times
- Useful as a prototyping tool
- Uses component technology
- Several emerging textbooks
- Language has
  1. Variable scope
  2. Dynamic arrays
  3. Functions and parameters
- Strong event-driven paradigm

Since this course will focus on problem decomposition, the user interface and file design will be provided by the instructor. Several projects will be given that force the student to develop some of their own rudimentary GUIs. The course will use flat file and relational data. Relational data will be provided by the instructor and will be in the form of PC based databases such as Microsoft Access. No instruction or use of SQL will be required for the first course.

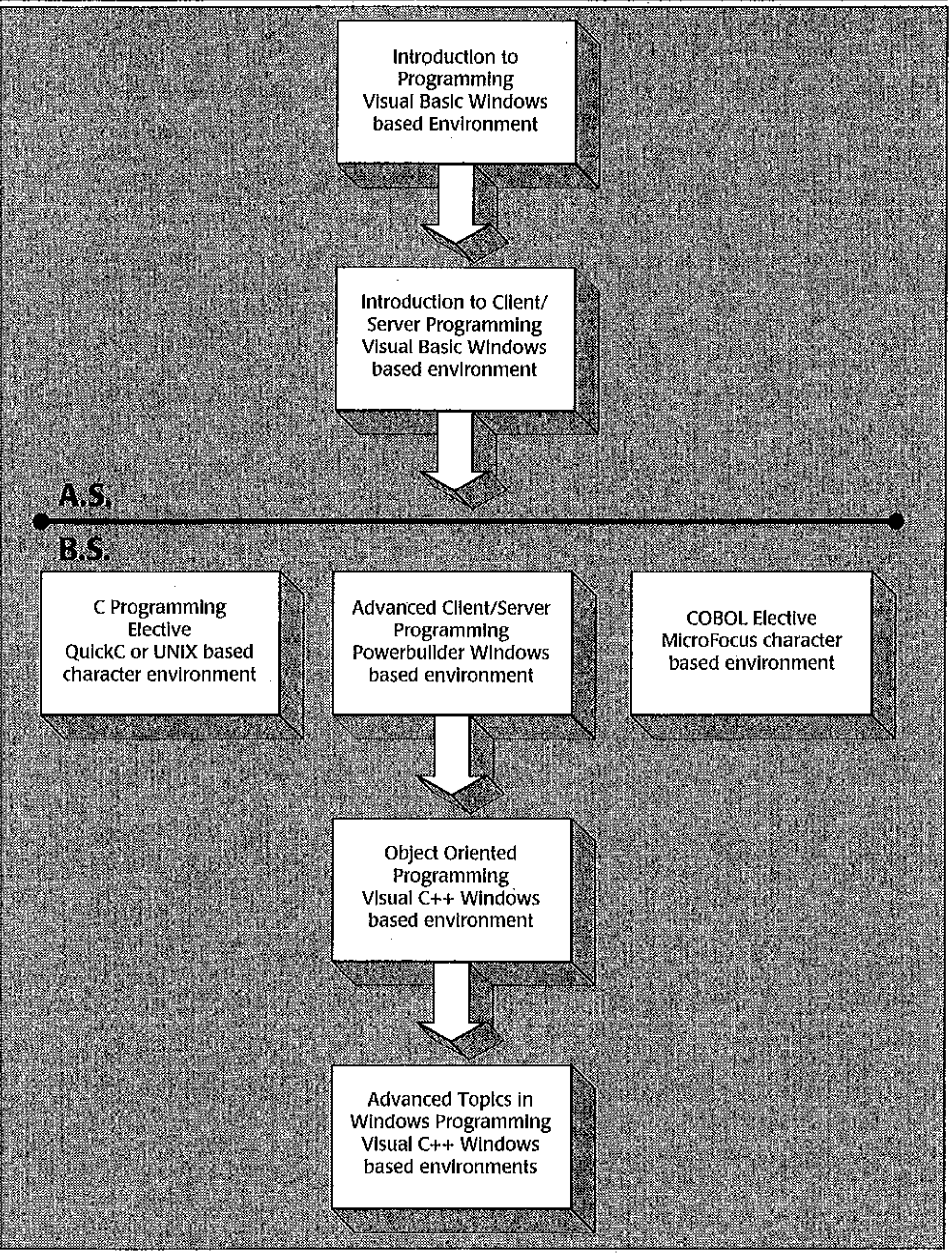
If object-oriented programming becomes more pervasive, then we may consider an alternative to Visual Basic for the first programming language. C++ and Smalltalk are currently the most dominant object-oriented programming languages, and according to Grady Booch (1994) this situation is not likely to change. He further asserts that Smalltalk is often used in the client side of client/server situations because it allows rapid development and adjusts easily to change. Because of the tight integration of Smalltalk and the GUI, we are studying the desirability of replacing Visual Basic with Smalltalk.

Introduction to Client/Server Programming

This course will expand on the previous course by introducing client/server programming. We feel it is important to introduce client/server programming in the associate degree curriculum. While most students matriculate to the baccalaureate program, students who terminate with an associate degree will be productive as programmers in current CIS organizations. Visual Basic Professional for Windows will be used for this course for two reasons: 1) strong database support and 2) student familiarity with the language. Topics to be covered in this course include:

- Database access using local, file server, and enterprise systems such as Oracle
- Use of ODBC and SQL for data access

FIGURE 2


Introduction to Programming Visual Basic Windows based Environment
Introduction to Client/ Server Programming Visual Basic Windows based environment
A.S.
B.S.
C Programming Elective QuickC or UNIX based character environment
Advanced Client/Server Programming Powerbuilder Windows based environment
COBOL Elective MicroFocus character based environment
Object Oriented Programming Visual C++ Windows based environment
Advanced Topics in Windows Programming Visual C++ Windows based environments

- Use of advanced features of enterprise systems such as stored procedures, triggers, and cursors
- Large group term project
- More extensive GUI design concepts

In this class all databases will be provided by the instructor. It is important to note that by this time in the student's career, each student will have completed a database literacy course that includes SQL. Some may argue that students can not handle information at this detail early on in their career, but after teaching a senior level course that compressed Visual Basic, event-driven programming, GUI Design, and client/server programming into a single semester with good results, we feel it can. Although, the senior level students suffered somewhat from having COBOL as a first language, new students will not experience similar difficulties. This is the last programming course of the associate degree program.

Advanced Client/Server Programming

This course will emphasize distributed database application development. Powersoft's Powerbuilder is an excellent tool choice for this course. Since Powerbuilder's language and approaches differ very little from Visual Basic, students will experience few difficulties in the transition to this tool. Powerbuilder was also chosen because of its new multi-platform capability in version four. Windows and Macintosh are the key information systems platforms to be covered. This course will be a logical extension to the introductory course with several other important topics to be covered:

- Distributed database access
- Security role enforcement
- Integration with other applications
- Network traffic analysis and optimization
- Emphasis on "shrink-wrap" quality code and design
- Multi-platform deployment

The neglect of the topics listed above combined with poor planning are many of the reasons client/server systems fail today. Traffic analysis may be the most important item in multi-user client/server systems. Poorly designed applications may result in hundreds of thousands of rows of a database returned to the client when only a few are needed. Today the Local Area Network (LAN) is the backbone of business operations, and downtime is very costly. A query that is acceptable at LAN speeds may not be at Wide Area Network (WAN) speeds. This type of business analysis will separate successful projects from failures.

Multi-platform deployment is also very important in open systems. In our department, Windows will be used as the development platform, but a few Macintosh and UNIX platforms will be available in the laboratofor cross development work and sting.

Students will be required to deliver hrink wrap quality programs in this ourse. This means applications ust include setup and installation atures. The design of help systems ill be deferred to the last course in e programming curriculum, dvanced Topics in Programming.

bject Oriented Programming

This course will emphasize object-iented programming and design. of this writing, C++ is being target-d as the language for this course. ++ was chosen because of it's erwhelming popularity, but we are searching the possibility of using a malltalk based visual programming vironment instead of C++, cause it is thought that Smalltalk is sier to use and enforces object inciples while C++ does not hillips, 1994). The first several eeks of this course will focus on + language constructs and object-ented programming. Because stu-nts will develop applications in the indows environment, heavy use of ss libraries will be required to old the complexity of the Windows I. During our design of this pro-amming track, we have reminded rselves that our curriculum is not a mputer science curriculum. A ver-n of this course has been proto-ed for two semesters with mixed ults. The students that enroll in the urse do not have the event-driven Windows background that stu-nts in the programming track will e, and suffer because of it. Topics his course include:

Abstraction principles
Persistence
Single and multiple inheritance
Polymorphism
Encapsulation
Object models
Class design

bject principles and class design the focal points for this course. Class design is currently missing from much of the curriculum, since it is inherently easier to teach the tools and technologies than the concepts. A foundation in object and class design will allow students to move to other object-oriented languages.

Advanced Topics in Programming

We believe it is necessary to require an advanced course which focuses on larger applications and team development. Emerging trends in application development will be taught in this course. Although Visual C++ will be the main development environment, other tool sets should also be utilized. Topics in this course include:

- Object Linking and Embedding (OLE)
- Application profiling
- Memory optimization
- Windows API usage
- Mixed language development
- Dynamic Link Libraries
- Reusability
- Component integration
- Revision control systems

Students will be required to work in teams where version control software is a necessity. Team projects will require individual members to define exact pragmatic interfaces for their code. This requires the design of reusable modules and libraries. Practical use of debugging tools will be utilized throughout the course.

COBOL Elective for Students not in Programming Track

Students not electing the programming track will have to take a COBOL elective as their third programming course. We were tempted to eliminate COBOL instruction entirely from the curriculum, but after reviewing industry trends we believe students not enrolled in the programming track should know COBOL for re-engineering purposes. In many cases legacy code is the only documentation for an application or system, and students need to be able to understand COBOL well enough to use re-engineering tools. Topics in this course include:

- COBOL
- VSAM
- CICS
- Embedded SQL

Important Concerns

Our choices of tool and language were not easily arrived at and may not be the absolute correct choice for another department. There are other competing languages which may be more suitable, but we tried to focus on the skill set and not on the implementing technology.

CONCLUSION

Transferable skills

We feel the following conceptual skill set can be obtained from this programming track:

1. Functional decomposition
2. Problem resolutions without technology
3. Structured Thinking
4. Structured Programming
5. Event-Driven Programming
6. Object-Oriented Design and Programming
7. Database front-end development - local, file server, enterprise
8. Distributed Application Development
9. Cross functional language knowledge awareness
10. Debugging Techniques
11. Testing Techniques
12. Development using team guidelines

It is important to realize that information systems educators will have to change the curriculum at a more rapid pace than previously. The technology and methodologies have outgrown the rate at which many CIS curricula can or are willing to change (Smith, 1994). COBOL was taught for a number of years as a first language. This document presents a rationale for teaching Visual Basic as a first language, but if something suits our skill set better in a year we may switch again. For example, we are currently debating whether object-oriented programming should be taught throughout the curriculum. If this occurs, then Visual Basic will no longer match our skill set, and we may change to Smalltalk, or C++ or possibly to Borland International's soon to be released Delphi tool. While this high rate of change represents a challenge to IS educators, the alternative of not changing will only result in a curriculum which becomes irrelevant to the IS organizations who we expect to hire our graduates.

AUTHORS' BIOGRAPHIES

John K. Gotwals is an Associate Professor of Computer Technology at Purdue University. He received his Ph.D. from Purdue University. His research interests include Object-Oriented Programming, Rapid Applications Development, and Programming to the Win32 Application Programming Interface.

Carlin Smith is an Instructor and Systems Administrator for the Department of Computer Technology at Purdue University. He is involved heavily in new technology investigation and application. His research interests include Object Oriented Applications Development, Client/Server Computing, and Multimedia applications development.

REFERENCES

Booch, G. (1994, November). Coming of age in an object-oriented world. *IEEE Software.* pp. 33-41.

Brumbaugh, L. (1994). Major revisions are overdue in introductory business programming courses. *Proceedings of the Eleventh Information Systems Education Conference,* 102-107.

Hensel, M. (1994). The approaching crisis in computing education: Enrollment, technology, curricula, standards and their impact on educational institutions. *Proceedings of the Eleventh Information Systems Education Conference,* 56-63.

Lauer, J. & Graf, D. (1994). COBOL: Icon of the past or symbol of the future? *Journal of Computer Information Systems, 34*(3), 67-71.

Newman, W. A. & Boman, B. (1994). Integrating visual event-driven languages into the MIS curriculum. *Proceedings of the Information Systems Educators Conference,* 175-179.

Phillips, R. (1994, October). Have you looked at Smalltalk lately? *Object Magazine.* pp. 14, 20.

Smith Jr., Carlin R. (1994). Introduction to Client/Server Application Development Using Visual Basic. *Proceedings of the Information Systems Educators Conference,* 195-196.

STATEMENT OF PEER REVIEW INTEGRITY

ISSN 1055-3096