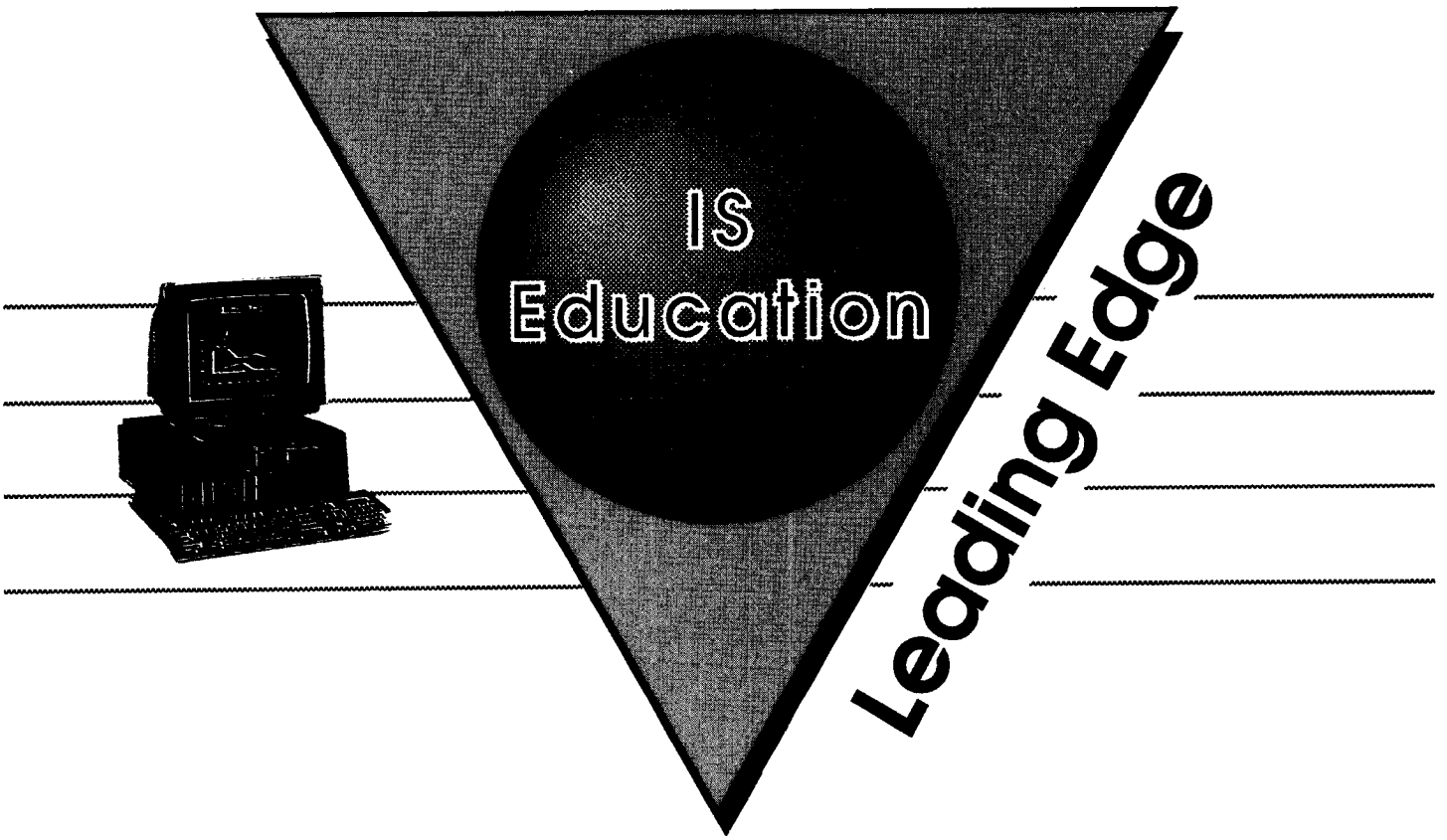
ISECON '92

October 17-18, 1992

Nashville Convention Center
Nashville, TN

Information Systems Education Conference



PROCEEDINGS OF THE NINTH INFORMATION SYSTEMS EDUCATION CONFERENCE

T.M.

DPMA

SPONSORED BY:
Data Processing Management Association
Education Foundation

ISECON '92

IS Education - The Leading Edge

On behalf of the Board of Regents of the Education Foundation of DPMA, the Executive Committee and the Track Chairs, I would like to thank you for your participation in ISECON '92.

While those of you who are teaching and publishing in the area of Information Systems might view this year's theme of "IS Education - The Leading Edge" as expounding on the obvious, I believe we sometimes need to be reminded of our rather unique position in education. As IS educators, we are inexorably linked to a discipline in which leading edge technological advances are made on a daily basis - and we have to try to keep up.

Because of your efforts, these Proceedings contain many quality papers that should help each of us move a little closer to the Leading Edge. As always, the included papers have been reviewed by a panel of your peers.

I encourage you to get the most you can from this conference by sharing your experiences and research with your colleagues from across the country.

Sincerely,

R. Wayne Headrick, PhD
Conference Chair

ISECON '92 STEERING COMMITTEE

CONFERENCE CHAIR

R. Wayne Headrick
New Mexico State Univeristy
Las Cruces, New Mexico

PROGRAM CHAIR

George Fowler
Texas A&M Univeristy
College Station, Texas

PROCEEDINGS CHAIR

Herbert Rebhun
University of Houston-Downtown
Houston, Texas

PLACEMENT CHAIR

George Morgan
Southwest Texas State Univeristy
San Marcos, Texas

EXHIBITS CHAIR

Sydney Rogers
Nashville State Technical Institute
Nashville, Tennessee

LOCAL ARRANGEMENTS CHAIR

Cary Hughes
Middle Tennessee State University
Murfreesboro, Tennessee

ISECON 1992

INDEX

SATURDAY OCTOBER 17

SESSION 1---10:30-11:45

SATURDAY

SESSION 2--1:30-2:45

SESSION 3--3:30-4:45

SUNDAY
OCTOBER 18

SESSION 4

SESSION 5--10:45-12:00

DEVELOPMENT AND USE OF INTERACTIVE MULTIMEDIA-BASED INSTRUCTIONAL DELIVERY SYSTEMS

Dr. D. V. Pigford and Dr. Greg Baur
Western Kentucky University

Abstract

This paper details an ongoing research project for the effective design and implementation of multimedia interactive computer aided learning (MICAL) in computer science. MICAL includes four progressive stages; Phase 1: Utilization of barcode controlled video; Phase II: Development of selected multimedia workstations; Phase III: Incorporation of expert system interfaces to the Level III applications; and Phase IV: Transfer of these applications to a microcomputer-based local area network.

This project is tailored to enhance the instructional efficacy of the Department of Computer Science and the quality of information delivery systems at Western Kentucky University.

Overview

Recent developments of multimedia technology is altering instructional delivery systems in both the business and academic environments. Using only a piece of chalk and a chalkboard has been compared to using "one stone age implement on top of another stone age implement" (Sneiderman, 1990). Combining graphics, video, audio, and computer technologies has established a new "Virtual Reality" and "Visualization" for learning environments. Implementing prototypes of a single user multimedia system is a step in the right direction, but it is a long way from assisting university lecture classes of forty or more students, or the needs of students who need more time and support to master conceptual learning. The objective of the MICAL research endeavor is "to refine, develop, and disseminate effective video-based multimedia systems for a wide range of students in a computer science department". The target audience at Western Kentucky University ranges from advanced graduate students in computer science to multiple sections of first time computer users. The delivery systems will first be utilized in the regular classroom, the department laboratories, and eventually the university laboratories. Common software and hardware platform(s) will be effected so that software/hardware implementations can be utilized, not just by the Computer Science department, but the entire university campus.

The main body of this paper discusses the four planned phases for the MICAL endeavor. The sections are as follows:

Phase 1:	Portable Barcode Controlled Videodisk Systems
Phase 2:	Level III Videodisk Workstations
Phase 3:	Expert System Interfaces for Video-based Workstations
Phase 4:	Local Area Networks for Video-based Workstations

The design constraints, environmental factors, and research questions for each phase will be highlighted in the rest of the paper. At the present time, the project has complete Phase I and is working on Phase II.

Phase I: Portable Barcodes Controlled Videodisk Systems

The initial start of MICAL was to select those existing videodisks most appropriate to computer literacy instruction and to incorporate their utilization in the standard lecture class of twenty to fifty students. The first phase did not include in-house production of videodisks (even though this is possible) due to the time, cost, and man hours of such an endeavor. The target audience for the initial phase was not undergraduate or graduates in computer science, but the literacy students in an introductory course in computer technology. The reason for this selection was the expanding enrollment in this required course (CS???). Currently there are multiple sections of thirty-five students per semester who study an introduction to microcomputer packages, programming, and social issues of technology. With limited resources and facilities, this heterogeneous lecture-oriented environment was the initial target audience.

The system selected for this scenario is a videodisk player with barcode controls for the instructor. The videodisk player (with Level III capability) is managed by scanning barcode symbols from a textual page to control the videodisk. Hence, the instructor was in complete control of the sequencing of the presentation. The system with its attached color monitor was mounted on a portable cart to permit easy portage to difference classrooms.

The videodisks utilized were those consistent with the course syllabus: history of computers, introduction to telecommunications, hardware design, human factors, and applications. Videodisks on microcomputer packages, other than DOS, were not utilized because of the potential change in versions of these software packages. The barcode development process included searching for frame numbers of video sequences with a hand control, printing barcode representation of videodisk instructions and frame numbers, and merging the barcode symbols with appropriate textual instructions using a wordprocessor. The lesson was created in consultation with the instructor, and refinements, changes, and resequencing were easily handled with a wordprocessor and barcode software.

Research questions included: Which video topics were most successful in this selected environment?; Why?; What human interface factors affected the instructor's utilization of this media system? Was the software development cycle for this technique easily transported to the instructor?; and Was there any correlation between the video topic and the learning style of the student? Finding answers to these questions was the main goal of Phase I along with the acceptance of the videodisk as another teaching tool for the professor.

Preliminary answers to the above questions produced some interesting data. First, it was found that there was a definite shortage of quality videodisk materials on the market for the level of the computer literacy student. Also, there was not an abundance of videodisk material available on more advanced topics for the level of a computer science major.

The initial videodisk chosen for this phase dealt with topics in computer literacy. Scripts were built around the various topics on the videodisk. The scripts contained

barcode symbols indicating appropriate frame numbers on the videodisk where appropriate still images and motion sequences were stored. The instructor was able to either use the prepared script of his/her own words to deal with the topic.

All of the chosen topics seemed to have about the same success when presented to students. Topics that had a combination of still frames and motion sequences were perhaps easier to script and kept student attention more easily, but this was not surprising. The limitations in scope of the topics available on the videodisk was a definite hindrance in answering the first question.

The instructors are currently in the process of using the system so little data is yet available. Initial reaction to using the system has been positive. Students have not yet used the system on a one-to-one basis so there is no data yet on their reactions to using the system.

Phase II: Level III Videodisk Workstations

Once videodisks have been utilized and integrated into selected classrooms, the most effective ones will be ported to Level III workstations. The computer control will be either a Zenith 80336 and/or Macintosh IIsi. This is consistent with the supported hardware platforms in the computer science department and consistent with the Zenith platform in campus-wide laboratories at Western Kentucky University. The software utilized will be Hyper-Card like packages which may include Windows 3.1, Toolbook, and/or Authorware. Even though the initial development for Level III Videodisk stations will follow a Macintosh platform, the Zenith will be incorporated. Software developed on the Macintosh with Authorware can be ported to the Zenith platform.

Since the computer science department has a Zenith/lcd display system in each classroom, a videodisk under computer control is viable. Both barcode and microcomputer controlled systems will be placed in the laboratories to supplement classroom instruction and promote individualized learning. Videodisk topics will expand to undergraduate and/or graduate components. Programming, software engineering, JCL, database, architecture, and similar topics will be included. Since many computer related topics are now being stored on compact disc read only memory (CD-ROM), this technology will be merged into the multi-media stand alone stations. Research directions will include software development for the multimedia components, effective graphical interfaces, appropriate help/testing components, and possible correlations between learning outcomes and participant learning styles. The effective incorporation of the microcomputer with video, overlay graphics, and CD-ROM will be the primary objective of Phase II (Pigford, 85), (Pigford, 84).

Phase III: Expert Systems Interface

Phase II development will be extended to expert system interfaces or expert front ends to the multimedia video workstations. An expert system is a software program that emulates the intelligence of a human expert in a narrow domain of expertise. An expert system will be used to control the interface to the video station, to diagnose students errors, to control sequencing in instruction and, in general, emulate the behavior of a master teacher with students in computer science. Video topics used for expert system development will be drawn from those topics which have proven to be most beneficial and efficacious in Phase I and II.

The software tools for expert system development must be incorporated with the authoring tools for the video workstation. Possible software solutions include VP-Expert, 1st Class, EXSYS, and Guru. VP-Expert is a viable candidate because of its integrability with existing software libraries (Pigford, 1990). Adding expert system interfaces and components will help to increase the efficiency and utility of the learning workstation, but the domain (video content) will tend to be narrower in scope (Baur, 1988).

The relevant research issues in this phase are designing intelligent user interfaces, establishing relevant knowledge bases for selected domains, testing the expert systems, and integrating the various software components of the multimedia workstation (Devolder, 1989), (Ting, 1988). Transforming the video workstation into an intelligent tutoring system is the central goal of Phase III.

Phase IV: Local Area Networks for Video-Based Workstations

Phase IV involves the expansion of the delivery system from essentially a single-user mode to a multiple-user mode. This extension will be accomplished by the use of multiple workstations connected with a local area network. The extension creates new problems. If the multimedia sources include a videodisk, then only a single user can use the material at one time. Hence, the concept of multiple users cannot be realized.

Second, if the multimedia source includes a CD-ROM, there are severe limitations. A CD-ROM system is not capable of reproducing full screen, full-motion video and has a limited capacity relative to the storage of still frames.

These limitations are caused by the fact that the storage of digital images (video) takes a great deal of memory which is not the case with video stored on a videodisk and is in analog format. The amount of storage needed for full-motion video sequences is so great that it is currently not feasible, even at high transfer rates, to move the images fast enough across a network to be able to reproduce them at a playback rate of 30 images per second, a rate that is required for full-motion video.

The purpose of the phase is to study and implement techniques that will permit the use of a local area network. A promising solution to the aforementioned problems is digital video interactive (DVI) which is under development, but the solution is still some time away.

Successful completion of this phase will enable students enrolled in computer science classes at Western Kentucky University to use the systems developed in Phase II and III of the project in a networked environment that will be cost effective.

Summary

This paper outlines a four phase research plan, MICAL, for developing a multimedia workstation environment for the Department of Computer Science at Western Kentucky University. The goal of the research effort is to improve the quality of instruction for different levels of students in computer science courses. The software and hardware platforms are designed to be ported to the university environment. The four phases involve the use of portable barcode videodisk configurations, Level III videodisk development, expert system interfaces, and transfer to local area networks. Constraints, environmental factors, and research questions for each of the four phases have been

discussed. The research plan follows the usual "re-evaluate and change" spiral found in the traditional software development process. A time frame of three years is expected, depending on the hardware/software developments for a networked implementation of the system.

References

Baur, G.R. and Pigford, D.V. (1988). Building an Expert System Using IVA Peripherals. Proceedings of the thirtieth International Conference of the Association for the Development of Computer-based instructional systems, 174-177, Philadelphia, PA.

Devolder, D.M. (1989) Design and Implementations of a Prototype Intelligent Tutoring System Incorporating Interactive Videodisc Technology. Unpublished masters thesis, Western Illinois University, Macomb, Illinois.

Hearn, D., and Baker M. (1986). Computer Graphics. Englewood Cliffs: Prentice Hall.

Helgerson, L. (1985). Buying graphic overlay for videodiscs. EITV. October 1985, 52-58.

Luthor, Arch. (1991). Digital Video Interactive. New York, Mc-Graw-Hill.

Pigford, D.V. (1990). Using an Expert System as an Authoring Tool for Interactive Videodisk Applications. Proceedings of the 32nd International Conference of the Association for the Development of Computer-Based Instructional Systems. pp 314-318, San Diego, CA.

Pigford, D.V., and Baur, G.R. (1990). Expert Systems for Business Concepts and Applications, Boston, MA: Boyd & Fraser.

Pigford, D. (1986). Graphics overlay for Interactive Videodisc Systems. Proceedings of the 28th International ADCIS Conference, p. 482.

Pigford, D.V., Henry, E.W., and Miller, A.F. (1985). Design of a Software Interface for an Intelligent, Interactive, and Integrated Videodisc System. Proceedings of the 26th International Conference of the Association for the Development of Computer-Based Instructional Systems, pp 157-164, Philadelphia, PA.

Pigford, D.V., Henry, E.W., and Miller, A.E. (1984). Assessing Integrated and Interactive Microcomputer Based Videodisc Authoring Systems. Proceedings of the Institute of Electronic and Electrical Engineers Educational Computing Conference, pp 79-86, San Jose, CA.

Sneiderman, B. (1990). Satellite Symposium: User Interface Strategies '91, University of Maryland.

Ting, Y. (1988). Design and Implementation of an Intelligent Interface to a Laser Videodisc System. Unpublished Master's thesis, Western Illinois University, Macomb, IL.

MULTIMEDIA:
THE ADVANTAGES AND THE DISADVANTAGES

Karen Ketler, Eastern Illinois University, Charleston, IL 61920 (217) 581-6646
John Walstrom, Eastern Illinois University, Charleston, IL 61920 (217) 581-6646

ABSTRACT

Multimedia is the computer-based combination of text, data, graphics, animation, music, speech, still images and full-motion video. It is the computer tool of the 90's. Multimedia business applications include 1) education and training, 2) business retailing, 3) 'edutainment' and 4) technical applications. Benefits include reduced costs, increased retention, increased sales, and better planning. Disadvantages include 1) hardware and software costs, 2) development difficulty, 3) speed of CD-ROM, 4) storage requirements, 5) copyright laws, and 6) lack of video compression standards.

INTRODUCTION

Multimedia is the computer-based combination of text, data, graphics, animation, audio (music and speech), still images, and full motion video that can be viewed from a computer monitor [5, 23]. What makes multimedia applications different from a movie is the ability to manipulate the digitalized forms of the images and audio on a personal computer. The types of organizations using multimedia and the applications of it are varied. But, there is a common thread that is woven through the different organizations and their different applications of multimedia--increased communication and retention. Average individuals only retain about 10% of what they hear, but up to 75% of what they see, hear and do [9].

It has been estimated that by the time a child enters school, he/she has watched 6,000 hours of television. That figure increases to 20,000 hours by the time the individual graduates from high school [1]. As a result, it is becoming increasingly difficult to stimulate these students with text-based materials. They are more interested in being entertained than in learning.

The State of Texas has recently decided that multimedia applications, such as videodisc-based materials, can be used in place of a traditional textbook. In keeping up with technology, Texas has adopted the videodisc, "Windows on Science" by Optical Data Corporation, as a science textbook [19]. Students are allowed to use the videodiscs to review lessons and to select options that interest them.

There is a direct relationship between the learning process and the business strategy. People only remember 10% of a verbal presentation after 10 days; however, 65% is retained after viewing a demonstration. Multimedia can make the learning process, both in schools and in business situations, more exciting and more effective.

THE USES OF MULTIMEDIA

The applications areas for multimedia can be divided into four categories [12]:

1. Education and Training. In this area the goal is self-paced learning, customized courses and tutorials.

2. Business Retailing. These applications include advertising in the form of customer presentations, product demonstrations, and merchandising kiosks.

3. Consumer Orientation. These applications center either on edutainment or presentations for the not-for-profit organizations. Travelogues, novels and games are examples of edutainment applications.

4. Technical Applications. In this area, multimedia is used in weather mapping, geophysical surveying, mapping, simulations, and dynamic computer-aided design and computer-aided manufacturing (CAD/CAM).

Education and Training

Education and training is one of the major business applications areas of multimedia. As stated earlier, the use of multimedia for training increases the retention rate. It is effective because it uses the talents of the best instructors [2]. BellSouth, which uses multimedia for training, estimates that the retention level has increased by 40%. But, the increased retention rate is not the only benefit; the task is also accomplished at a reduced cost. BellSouth estimates the cost savings as high as 80% [20]. An IBM plant in Poughkeepsie is using multimedia for training and is saving $800,000 [11].

GTE North in Westfield, Indiana is another organization which uses multimedia as a training tool. The multimedia application provides deeper understanding than a lecture and the retention rate is 3 to 4 times higher. The multimedia application is a lesson on how to fix telephone cables. Various sounds can be heard, and pictures can been seen. The user can zoom in on pictures to examine details such as a needle on a meter or to look beneath a manhole cover. This multimedia application makes training equivalent to "hands-on" experience [17].

A third advantage, as seen at the University of Illinois, is the elimination of hazardous training procedures. University of Illinois is one educational facility that has been using multimedia for over five years. A chemistry interactive videodisc laboratory was developed in 1986 to assist introductory chemistry students with required coursework. Chemical experiments which were too hazardous, expensive, or time-consuming to physically be conducted are demonstrated in the videodisc laboratory [18].

Business Retailing

The implementation of multimedia in business retailing, such as kiosk applications, is a logical progression from the Automatic Teller Machines (ATM). The key is how to present information to a customer that will increase the likelihood that the customer will make a purchase.

Multimedia eliminates detailed training of the sales staff. The application will communicate all the details of the product to the customer. No longer will a customer be turned off by the absence of a salesperson or by the lack of knowledge of the salesperson.

Mannington Mills, a flooring manufacturer, has developed a multimedia application which allows a customer to select a kitchen, bathroom, foyer or family room design similar to the room in his/her own home, The customer can then view the room using 300 different colors of flooring and 15 different arrangements of furniture. The ultimate objective is to show the prospective buyer what a difference new flooring can make in a room. In addition to viewing the room, the customer can obtain information about products, installation, guarantees and floor care. This multimedia system also maintains statistics on how many customers have looked each color and pattern of flooring as a means of assisting the retailer in their purchasing and inventory decisions [14]. Some retailers have indicated a 100% increase in sales since they began using this application [23].

Buick Motor Division is starting to use advertising on a disk. These disks were mailed to prospective customers and to dealerships. The multimedia presentation allows a customer to browse through the description and animated pictures of Buicks by using a PC. The presentation includes the sounds of the engines. Users can select the model and the options that appeal to them. Then, they may select the routine which, through the use of a spreadsheet, will calculate the cost and the estimated payments. The user also has the option of comparing the Buick with a competitor's car.

As a result of this multimedia advertising, Buick Motor Division is convinced that the customer spends more time with the personal computer application than with the brochure. Approximately 12% of the individuals with the multimedia application have purchased a Buick. This is about doubled the normal response from Buick advertising campaigns [17].

Consumer Orientation

The most common application of multimedia in the consumer orientation area is games. However, this group is not all entertainment related. With the changes in the tax laws, many charities are looking to multimedia as a method of increasing their charitable contributions. The American Heart Association is one such organization which is using multimedia. The highlight of this multimedia application on how to manage blood cholesterol is a 5 minute computer-animated cartoon [17].

A more unique application is the development of the 19 miles of Mississippi riverfront property in the St. Louis area. The goal is proper environmental planning. The application was developed through funding by the National Endowment for the Arts. Information about the environment, history and land use of various sites along the Mississippi River were included in the application. A user can select a site by 1) using key words and allowing the computer to recommend a site, 2) pointing to a location on a map, or 3) using an aerial view to locate a site. The user can browse through various developmental sites. Information about the site, a video picture, and an aerial view is given to the user. The developers hope that the system will be used in the planning process [24].

Technical Applications

An example of a technical application of multimedia is the architect who leads a customer on a walking tour of a new building. The customer is allowed to wander through the building at his/her leisure. It is not a problem if the customer requires radical changes. It is done almost instantaneously because the tour was a multimedia presentation of the building plans. This CAD/CAM

application is a variation of "virtual reality" which has been implemented in games [8].

TECHNOLOGY OF MULTIMEDIA

Multimedia technology has three elements: 1) the computer hardware, 2) a large storage device (CD-ROM, for example), and 3) the multimedia software.

Computer Hardware

Both the IBM PS/2 (or 100% compatible) and the Apple Macintosh computers are excellent choices for multimedia systems. The hardware question, whether to use IBM or Apple, currently limits what software options are available to the user. However, IBM and Apple have recently reached an agreement to 1) share hardware technology, 2) jointly develop a new microprocessor using RISC technology and 3) form a jointly owned company to develop multimedia software [3]. But, for today, one must decide whether to use IBM or Apple.

IBM PS/2 System

An IBM PS/2 or compatible computer with a 386 or 486 chip, a built-in CD-ROM drive, a graphical user interface (GUI) such as Windows 3.0, a minimum of 2 megabytes of memory, a 40 megabyte hard disk, a mouse, and 4-bit color graphics at VGA resolution is recommended for multimedia applications [16]. Additional accessories must be added to the basic computer hardware. A fully equipped IBM system for multimedia will cost approximately $20,000.

The **M-Motion Video Adapter/A**, which sells for approximately $2,250, allows full-motion video and audio on the system. Video is digitized 'on the fly' and can be displayed full-screen or in multiple windows. Video can be obtained from a videodisc, a video camera, closed circuit television, or a VCR. While this adaptor provides for analog video stored on videodisc or VCR tape, the **ActionMedia Capture and Display Adapters** allow for storage of the digitalized video on a hard drive or optical disc. The optical drive, **PS/2 Rewritable Optical Drive**, is a 3.5 inch optical storage device with 127 megabytes. It resembles a diskette. For digitalization of still pictures from a video camera or a videodisc player, the **Video Capture Adapter/A**, which also retails for $2,250, is needed. Once a picture is digitalized, it can be displayed on any CGA monitor [7].

Sound can be digitalized with the **M-Audio Capture and Playback Adapter/A**, which retails for $565. It provides not only for the digitalization of 16-bit stereo sound, but also provides either storage saving compression or sound quality optimization. There is a tradeoff between storage compression and sound quality. **Digispeech DS 201** combines a speaker and electronics into a single, highly affordable unit for recording and playback of speech and music [4].

Apple Macintosh Hardware

For multimedia, a Macintosh II series or the Quadra series is required. The Motorola 68030 or 68040 is recommended. With the Macintosh, the graphical interface and stereo sound are built-in; thus, additional hardware is only required for the use of motion video. A RasterOps 364 board that sells for $640 allows the use of motion video [4].

Large Storage Devices

Compact Disc Read-Only Memory (CD-ROM) has been a major factor in the rapid growth of multimedia. A CD-ROM disk, retailing for approximately $10, can store 600MB or over 300,000 pages of text. This 4.72 inch platter can be search in a few minutes. Since this is considerably longer than the time it takes to search a hard disk, it is not surprising to realize that users of CD-ROM complain about its speed.

A second complaint is about the price of a CD-ROM drive. These drives in the past have retailed for as much as $1,500, but recently the prices have dropped to under $500. Industry experts are predicting that the price of CD-ROMs must drop to approximately $300 before the rapid growth of multimedia can be realized [13].

A third complaint about CD-ROMs deals with video storage and compression. In order to store a single-screen at low resolution 300 kilobytes of data are required. Full-motion video displays 30 frames each second. Therefore, one second of motion would require nine megabytes of data storage. Sixty seconds of full motion video is generally beyond the capacity of most desktop computers. Storage compression is a must [22]. However, storage compression leads to the fourth disadvantage. There is no standard data compression techniques. In addition, compression may slow down the retrieval times.

Multimedia Software

There are several options for multimedia software; however, the options are limited by whether the hardware is IBM (or compatible) or Apple.

IBM Software

LinkWay is a multimedia software tool designed for use in an educational setting. A first grader can learn this

authoring tool with only 15 minutes of training. The user selects the still pictures, digitalize them, and adds appropriate text [10]. LinkWay also allows the user to combine color, graphics, music, voice and full-motion video with the still pictures and text [7].

Storyboard Live!, which retails for $495, is another multimedia software product for the IBM Personal Computer. It gives the user the ability to design anything from simple slide shows to complex interactive presentations containing animation, voice, music, and full-motion video. However, the motion video sequences are limited to a 4 inch by 4 inch size. Users can quickly learn how to make a simple presentation with Storyboard Live!. As their expertise increases, they can easily learn to add the other functions. In a recent evaluation by InfoWorld, Storyboard Live! received a very good rating [5].

The top-of-the-line software product for IBM machines is the **Audio Visual Connection (AVC)**, which also retails for $495. It is a "powerful award-winning authoring system that assists in the creation of complex, professional-quality multimedia courseware and presentations complete with high-resolution visuals and stereo audio" [7]. It is geared to the professionally trained multimedia specialist.

Apple Macintosh Software

The **HyperCard** by Apple is a multimedia tool which comes with every Macintosh machine. Thus, it is not surprising that it is one of the most widely used multimedia tools. It is a simple tool, designed for the non-professional. It is very popular is educational settings. Like Storyboard Live!, users can start out with simple presentations and as their expertise increases, they can easily add the other functions.

The more professional multimedia tools available for the Macintosh includes **Macromind Director** at $995. InfoWorld recently rated Macromind Director higher than Storyboard Live! on the IBM. This package, introduced in 1989, has two modules: Studio and Overview. The studio module allows the user to create text and animation sequences while the overview module is used film clips, sound sequences, and special transition effects [6].

The newest product for the Macintosh is **QuickTime** by Apple, which retails for only $350. It enables the user to "store time-based data - video, animation, and sound, on a hard drive and quickly attach it to almost any document" [13]. The two major advantages of QuickTime are: 1) one can view a multimedia presentation without any special hardware; and, 2) since all media is digitalized, it can be sent over an Ethernet network or by E-Mail [4].

Design Personnel

The effective use of technology is usually dependent upon the skills of the individuals using it. Although supply of qualified individuals is a problem throughout the computer industry, it is very evident in the emerging area of multimedia. Very few individuals have all the skills of scripting, writing, graphics, video and computer programming.

ADVANTAGES AND DISADVANTAGES

Each application of multimedia has its own specific objectives and desired benefits. The benefits are as numerous as the applications. These benefits include:

1) Reduced Costs
2) Increased Retention
3) Increased Information
4) Increased Consumer Attention
5) Individualized Learning
6) Increased Sales
7) Reduction in Hazardous Training
8) Better Planning
9) Increased Interaction between User and Application

Multimedia is a new technology, and with any new technology, there are problems which need to be resolved. A major disadvantage is storage requirements and associated compression. The three compression techniques currently used just add confusion to the new multimedia user. Users are hesitant to spend money on multimedia hardware and software until the Joint Photographers Experts Group (JPEG) decides which one of the three compression techniques is going to be the industry standard. Other disadvantages include:

1) Hardware Costs
2) Software Costs
3) Supply of Technical Professionals
4) Difficulty in Development
5) Speed of CD-ROM
6) Copyright Laws

BIBLIOGRAPHY

[1] Braun, M. "Personalizing the Information Revolution." T.H.E. Journal (Supplement). September, 1991, p. 1.

[2] "Closing the Skills Gap." Multimedia Solutions. May/June, 1991, pp. 3-5.

[3] Depke, D. A. and Rebello, K. "IBM-Apple Could be Fearsome." Business Week. October 7, 1991, pp. 28-30.

[4] Gibson, B. Multimedia. Unpublished Working Paper. Eastern Illinois University. November, 1991.

[5] Green, D. and Green, D. "IBM's Storyboard Live Offers Expanded Multimedia Tools." InfoWorld. February 25, 1991, pp. 66-69.

[6] Green, D. and Green, D. "MacroMind Director Take 3: Same Cast, Better Action." InfoWorld. August 5, 1991, pp. 89-92.

[7] "IBM Multimedia Family of Products." T.H.E. Journal (Supplement). September, 1991, pp. 31-32.

[8] Johnston, S. J. "Multimedia: Myth vs. Reality." InfoWorld. February 19, 1990, pp. 47-52.

[9] Keyes, J. "Multimedia Offers Managers Multiple Business Solutions." ComputerWorld. September 24, 1990, p. 112.

[10] "Model Applications." T.H.E. Journal (Supplement). September, 1991, pp. 6-9.

[11] "Multimedia: Up Close and Personal". IBM Directions. Spring, 1991, pp. 2-7.

[12] Murphy, J.A. "Multimedia is today's message." Today's Office. February, 1990, pp. 6-14.

[13] Poole, L. "QuickTime in Motion." MacWorld. September, 1991, pp. 154-159.

[14] "Real-World Applications." T.H.E Journal (Supplement). September, 1991, pp. 29-30.

[15] Romei, L. K. (editor). "Multimedia: Delivering Added Value Today". Modern Office Technology. October, 1991, pp. 43-46.

[16] Rosen, L. "Sorting Through the Multimedia Maze." Information Today. September, 1991, pp. 19-21.

[17] Shao, M. and Brandt, R. "It's a PC, It's a TV--It's Multimedia." Business Week. October 9, 1989, pp. 152-166.

[18] Smith, S. G. and Jones, L. L. "The Acid Test: Five Years of Multimedia Chemistry." T.H.E. Journal (Supplement). September, 1991, pp. 21-23.

[19] Soloway, E. "How the Nintendo Generation Learns." Communications of the ACM. September, 1991, pp. 23- 26, 95.

[20] "The Art of Listing at BellSouth." Multimedia Solutions. May/June, 1991, pp. 14-16.

[21] Wallace, S. "Desktop Spectaculars". CIO. October, 1990, pp. 114-120.

[22] Wallace, S. "Setting the Scene". CIO. October, 1990, p. 120.

[23] Wiegner, K. K. and Schlax, J. "Showtime." Forbes. July 22, 1991, pp. 294-296.

[24] Wiggins, L. L. and Shiffer, M. J. Planning with Hypermedia: Combining Text, Graphics, Sound, and Video." Journal of the American Planning Association. Spring, 1990, pp. 226-235.

TEACHING CASE TOOLS FOR EXISTING SYSTEMS

Alden C. Lorents
Northern Arizona University, CBA 15066, Flagstaff AZ, 86011

ABSTRACT

The decade of the nineties will see a lot of activity in reengineering existing COBOL based systems into newer COBOL based systems, CASE driven COBOL systems, or newer 4GL systems. Much of this activity will be supported by CASE for existing systems that will aid in the analysis and reverse engineering of existing systems. Students in CIS curriculums should be exposed to the methods and tools for doing software maintenance, reverse engineering and systems reengineering. Most curriculums do not teach methods of migrating existing systems into newer systems, often supported by newer database technologies. This paper discusses some of the components of CASE tools for existing systems, and looks at some ideas of incorporating these tools into the curriculum.

INTRODUCTION

Most curriculums in CIS over the years have concentrated on the tools, techniques, methodologies, and technologies, related to the development of new applications. Very little is taught related to the maintenance, enhancement, or evolution of existing systems. This is partly due to the fact that most development tools tend to be independent, and not easily integrated with other development tools, especially CASE tools for existing systems. This environment is beginning to change. CASE tools are being built that analyze existing systems, capture the data and process definitions, and provide diagrams and reports that support the modification, enhancement, and reengineering of an application. Eventually these reverse engineering tools will be able to communicate with forward engineering tools to make the process of evolving systems into the next generation much easier.

Various articles estimate that there are about 70 billion lines of COBOL code in operation with about 5 billion lines added each year [Eliot]. There are approximately 2 million COBOL programmer/analysts maintaining and developing these systems. Eventually all these systems will be reengineered. In some cases the systems will be completely replaced with new technologies using 4GL application development tools or CASE tools with 3GL generators. In other cases the systems will be replaced with packaged systems. In the majority of the cases, the systems will continue to evolve with parts of the system being rebuilt. All of these approaches have one thing in common...the analysis of the existing system. Reverse engineering the rules, formulas, data structures, data constraints, and processing algorithms is a key function in the maintenance and evolution of existing systems.

Students in CIS programs should be given some exposure to working with CASE tools for existing systems. There are various reasons for this. a) Most entry level positions are involved with maintaining existing systems [Ridgeway], b) there will be more activity in the future to reengineer existing systems, as CASE tools and standardized repository systems are developed to support this activity [Bush, Maxson, McMullen, Snell, Gunn], c) students can learn a lot about digging into a complex older system.... like medical students working with a cadaver.

The following is a discussion of some CASE tools for existing systems, and some approaches to using these tools in the CIS curriculum.

CASE TOOLS FOR EXISTING SYSTEMS

There are a number of tools on the market that aid in the process of software maintenance and reengineering. This paper discusses three tools that have a fair amount of reengineering functionality, and then illustrates how one of the tools can be used to introduce students to these concepts. Micro Focus Workbench and Intersolv Design Recovery are tools that are currently available for the PC/DOS environment. The VIASOFT Existing Systems Workbench will be available for the OS/2 environment in the next year. PC based tools are normally easier for educational institutions to obtain and maintain for curriculum support. Micro Focus also supports an excellent academic grant program.

MICRO FOCUS WORKBENCH

The Micro Focus Workbench (MF WB) is a full COBOL development workbench that includes tools for new development and for existing systems. The WB includes structure and analyzer components, and a COBOL Source Information (CSI) component that is a type of repository. CSI provides various query information to the COBOL smart editor, and the powerful online debugger called animator. The analyzer supports information on test paths and code that has or has not

been tested. Academic programs that currently have this product under the Micro Focus academic grant program already have a number of tools to teach many of the concepts related to software maintenance, reengineering and reverse engineering. Some of the features supported by the MF workbench are as follows:

1) A top down structure diagram can be created, printed and/or referenced online. The diagram is coordinated with the source code so as you move around the structure diagram window with a mouse, the related source code will scroll within the background window.
2) Procedure names can be pointed to anywhere in the code, obtaining a window showing which routines execute the specified procedure, and which routines are executed by the procedure.
3) Data names can be pointed to anywhere in the code, obtaining a window showing where the data is used and where it is defined. You can also find out if the data is a part of higher level data grouping, and the various references in the program to that higher level grouping.
4) Queries can be done on any command or group of commands such as I-O or arithmetic that isolate all code in the program with those commands.
5) Dead data and potential unused procedures can be isolated and reviewed for possible removal from the program.
6) Reports or windows can be obtained to show various program dimensions (metrics).
7) There is a complete support system to manage computer aided online testing and test systems.

INTERSOLV DESIGN RECOVERY TOOL

Design Recovery DOS is a recent tool that has been introduced by Intersolv as a part of their development tools package. Other tools include Excelerator, Application Development System (APS), prototyping (DEMO II), and configuration management (PVCS). Design Recovery supports COBOL 85, IMS, IMS MFS, CICS and DB2 environments. All process and data models extracted from code are stored in Intersolv's repository. Some of the functionality supported by the product includes:

1) Creation of a data structure diagram of the program.
2) Capture of all IMS data definitions (DBD's).
3) Capture of all COBOL data definitions in the program
4) Creation of all screen designs from IMS MFS and CICS code.
5) A relational modeler that captures VSAM, IMS, and DB2 system definitions and does forward engineering to DB2 and other relational databases.
6) Various reports are available to show both data and process relationships.
7) The calculation of a Cyclic Complexity Index that is a measure of the number of logic paths in a paragraph, section, or program.

Intersolv's Excelerator is available through an academic grant program. The Design Recovery Tool is not currently available under that same grant program.

VIASOFT'S EXISTING SYSTEMS WORKBENCH

Viasoft is a company that specializes in CASE tools for existing systems. The products VIA/SmartEdit, VIA/Insight, VIA/SmartDoc, VIA/SmartTest, VIA/Recap and VIA/Renaissance are all products that support software maintenance and the reengineering of COBOL programs. VIA/Insight is a analyzer that gathers comprehensive information about a program which it stores in a repository. This allows developers to look at all the process and data relationships of a program with powerful retrieval commands. The product supports various logic flow and data analysis similar to the Micro Focus Workbench. VIA/Smartdoc produces reports on program structure, and generates various metrics. VIA/Smartedit is a COBOL smart editor that knows the process and data relationships in the program. VIA/SmartTest supports the management of a test system and the analysis of how well a program has been tested. It also has a powerful online debug facility.

VIA/Renaissance is a recent product comprised of three components which support the various phases of re-engineering: analysis, code extraction, and forward engineering. It is a product that specializes in the decomposition of large COBOL programs into smaller more manageable units. The product stores the decomposition and analysis in a repository which can be accessed by the other VIASOFT products. The combination of these products makes up what VIASOFT refers to as its Existing Systems Workbench (ESW). The product supports all of the typical analysis tools and also supports some forward engineering capability of generating new programs that consist of parts of the old programs that can stand alone. It also can provide information that will link to other forward engineering CASE tools.

USING CASE TOOLS FOR EXISTING SYSTEMS IN THE CURRICULUM

There is currently very little text support on introducing software reengineering, software maintenance, or reverse engineering concepts to students. Projects and cases need to be developed using some of the tools in the CASE products mentioned above. The following is a discussion of a sample project, some analysis that can be done using the tools, and some ideas on how to present this topic to students. The CASE tool used to illustrate these examples is the Micro Focus Workbench.

Projects can be set up as part of the advanced COBOL course, the analysis and design course, or as part of the advanced projects course. In an advanced projects

course, a project could be set up to include a number of older COBOL programs that are part of some subsystem. These programs could be obtained from the school or from local industry. Projects can be set up to reengineer a part of the system. This involves analysis, redesign, modifications, and implementation. Projects could also be setup to just rework a COBOL program to make it more maintainable with no new functionality.

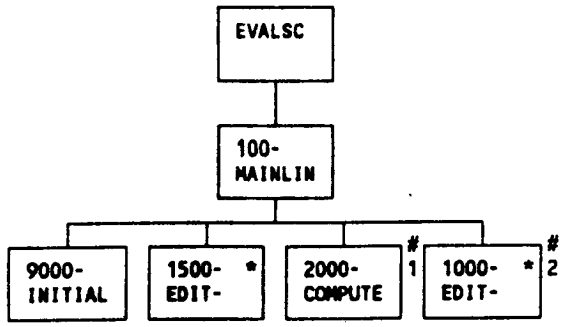
SOME SAMPLE ANALYSIS TECHNIQUES USING MICRO FOCUS WB

The program used for these examples is a current production program used to produce faculty evaluation reports. The sample analysis highlights some of the features of the tool, and some ideas relative to its use by CIS majors.

The program used in these samples was downloaded from the mainframe to the PC, compiled by the MF checker, with all the structure and analyzer features on. This process created a COBOL Source Information File (CSI) that supports various analysis features in both the editor and online debugger (Advanced Animator).

Structure Diagram. Students can start by doing some general analysis of the program to get an initial understanding of it. A top down program structure diagram is produced for the entire program that can be printed or used in the animator. The animator allows you to jump from box to box in one window while the associated code for that box pops up in another window. A sample output of this feature is shown in figure 1.

FIGURE 1.



Metrics. A summary of various metrics can be obtained to get an idea of the size, complexity, and a feel for the maintainability of the program. A sample of this output is shown in figure 2. The program volume figure is a metric that measures program complexity and is derived from the formula, Volume = (N1+N2)LOG2(n1+n2) where N1=Number of statements, N2=Number of references to all identifier names, n1=number of unique verb types, and n2 = number of identifier names. The metric is most useful in assessing the relative complexity between programs.

FIGURE 2.
SUMMARY OF PROGRAM DIMENSIONS

Statistics		Verb counts	
Lines of Source	1642	Compute	36
Lines of Code	1102	Move	257
Comment lines	61	Add	23
Comments /100 lines	3	Subtract	2
Statements	560	If	68
Sections	12	Goto	39
Paragraphs	62	Perform	67
Files	3	Exit	11
Data-items	162	Open	3
Condition	7	Close	2
Screen Name	0	Read	4
Report	0	Write	4
Level 78	0	Accept	1
Call (by number)	0	Next	8
Call (by name)	0	Else	28
Call (data-item)	0	Examine	4
		Perform thru	2
		Stop Run	1
		I-O	13
Program Exits	1		
Arithmetic	61		
Maximum Nesting	3		
Overlapping performs	0		
Program Volume	14128		

Isolating I-O. Students can diagram the inputs and outputs of the program by looking at the I-O references within the program. CSI can display window that is the result of a query on a keyword I-O as shown in figure 3. The query isolates all I-O statements and their location in the program. The student can view each statement along with the related code in that routine, or view just the statements in compressed mode as shown in figure 3.

FIGURE 3.
QUERY REPORT ON I-O STATEMENTS

```
                                        561 lines unshown
562   OPEN INPUT OPSCAN-TEMP.                    <= = = = Use
                                        4 lines unshown
567   READ  OPSCAN-TEMP INTO WS-OPSCAN-TAPE  <= = = = Use
                                        8 lines unshown
576   CLOSE OPSCAN-TEMP.                         <= = = = Use
                                        7 lines unshown
584   OPEN OUTPUT REPORT-OUT.                    <= = = = Use
                                        3 lines unshown
588   OPEN OUTPUT CLASS-REPORT-OUT               <= = = = Use
                                        6 lines unshown
595   READ OPSCAN-TEMP INTO WS-OPSCAN-TAPE   <= = = = Use
                                        7 lines unshown
603   CLOSE  REPORT-OUT  CLASS-REPORT-OUT  OPSCAN-TEMP.  <= Use
                                        60 lines unshown
664   READ  OPSCAN-TEMP INTO  WS-OPSCAN-TAPE  <= = = = Use
                                        62 lines unshown
727   READ OPSCAN-TEMP INTO WS-OPSCAN-TAPE   <= = = = Use
                                        880 lines unshown
```

One exercise could be to move all I-O operations such as READ and WRITE to an I-O utility section of the

program. Reads of any file should be consolidated to one actual READ using performs. The I-O report can be used to find the read statements. Other analysis can also be done to verify references to the read operation. A query can be done on a data area in working storage. The query will show all references where the data is modified as shown in figure 4.

FIGURE 4.
QUERY REPORT ON ALL READ STATEMENTS

```
Data----------------------size-refs----------------------------------------
WS-OPSCAN-TAPE Group          66   6m
Working-Storage 01/77-level
------------------------------------------------------------------------
                                        401 lines unshown
 402 01  WS-OPSCAN-TAPE.                                  <====Defn
                                        164 lines unshown
 567     READ  OPSCAN-TEMP INTO WS-OPSCAN-TAPE   <====Mod
                                         27 lines unshown
 595     READ OPSCAN-TEMP INTO WS-OPSCAN-TAPE    <====Mod
                                         66 lines unshown
 662     MOVE SPACES TO WS-OPSCAN-TAPE.          <====Mod
 663                                     EVA06630
 664     READ  OPSCAN-TEMP INTO  WS-OPSCAN-TAPE  <====Mod
                                         62 lines unshown
 727     READ OPSCAN-TEMP INTO WS-OPSCAN-TAPE    <====Mod
------------------------------------------------------------------------
```

Procedure Flow. Another approach is to query a specific procedure name and obtain all references where that procedure is executed from, and references to procedures it executes. The procedure query is shown in figure 5. Data or procedure queries are easy to do with mouse support. You can point to a data name or a procedure name, double click on it and get a display of all references.

FIGURE 5.
QUERY ON A SPECIFIC PROCEDURE

```
Procedure-----------------------------------------------------------------
1400-READ-A-RECORD Paragraph
==================================
Executed from:-
  1100-IS-IT-HEADER  GO TO
  1100-IS-IT-HEADER  GO TO
  1200-IT-IS-STUDENT GO TO

Executes:-
  GO TO       1499-EXIT
  GO TO       1100-IS-IT-HEADER
------------------------------------------------------------------------
                                        619 lines unshown
 620                GO TO 1400-READ-A-RECORD          <==EFrom
                                        8 lines unshown
 629                GO TO 1400-READ-A-RECORD.         <==EFrom
                                        8 lines unshown
 638     GO TO 1400-READ-A-RECORD.                    <==EFrom
                                        18 lines unshown
 657 1400-READ-A-RECORD.                              <====Defn
                                        2 lines unshown
 660       GO TO 1499-EXIT.                           <====Exec
                                        7 lines unshown
 668             GO TO  1100-IS-IT-HEADER.            <====Exec
------------------------------------------------------------------------
```

Consolidation. Other analysis can be done by the students to try to improve maintainability. There may be a potential to consolidate duplicate logic in the program. A query can be done on all arithmetic operations, or on any specific command such as COMPUTE. An example of a query on arithmetic statements is shown in figure 6.

FIGURE 6.
SAMPLE OF ISOLATING COMPUTATIONS

```
 1156    ADD 0.005 TO CLASS-SD                  <====Use
                                        63 lines unshown
 1220    COMPUTE CLASS-SIGX  = CLASS-SIGX       <====Use
                                        8 lines unshown
 1229    COMPUTE CLASS-SIGX2  = CLASS-SIGX2     <====Use
                                        8 lines unshown
 1238            ADD QCT-RESP-DATA (SUBSC, SUBSC2) TO ABS-TOT-R.
<====Use
                                        32 lines unshown
 1271              ADD TRUNCATED-DATA (SUBSC2) TO HOLD-CUM-T
<====Use
                                        5 lines unshown
 1277       COMPUTE HOLD-REL ROUNDED =          <====Use
                                        6 lines unshown
 1284      COMPUTE HOLD-REL-T ROUNDED = (TRUNCATED-DATA (SUBSC2)
/ <====Use
                                        6 lines unshown
 1291      COMPUTE TOT-REL-R ROUNDED = TOT-REL-R + HOLD-REL.
<====Use
 1292                                   EVA12920
 1293      COMPUTE TOT-REL-T ROUNDED = TOT-REL-T + HOLD-REL-T.
<====Use
                                        19 lines unshown
 1313       ADD TRUNCATED-DATA (SUBSC2) TO      <====Use
                                        105 lines unshown
 1419    ADD 1 TO WS-PAGE-COUNT-G.              <====Use
                                        125 lines unshown
 1545    ADD 1 TO WS-PAGE-COUNT-C.              <====Use
                                        66 lines unshown
 1612    ADD WS-LINE-SPACE-G TO WS-LINE-COUNT-G. <====Use
                                        15 lines unshown
 1628    ADD WS-LINE-SPACE-C TO WS-LINE-COUNT-C. <====Use
------------------------------------------------------------------------
```

Structure Enhancement. Another possible exercise would be to clean up some of the unstructured GO TO programming. This can be aided by querying the various routines with a query on the key word GO TO. An example is shown in figure 7. The students could use this analysis to regroup routines by performing them and eliminating multiple exits from a routine.

FIGURE 7.
QUERY REPORT ON GO TO STATEMENTS

```
                                        612 lines unshown
 613        GO TO 1200-IT-IS-STUDENT                 <====Use
                                        6 lines unshown
 620              GO TO 1400-READ-A-RECORD           <====Use
                                        8 lines unshown
 629              GO TO 1400-READ-A-RECORD.          <====Use
                                        8 lines unshown
 638     GO TO 1400-READ-A-RECORD.                   <====Use
                                        21 lines unshown
 660       GO TO 1499-EXIT.                          <====Use
                                        7 lines unshown
```

General Cleanup. Other cleanup can be done by analyzing potential dead code through a query of non referenced procedures, and a query on dead data. Figure 8 shows a sample report on dead data and figure 9 shows a sample report on potential dead code. Analysis needs to be done on the non referenced procedures, because not all non referenced procedures are necessarily dead code.

FIGURE 8.
QUERY REPORT ON DEAD DATA

Summary Report

Dead-data	size	
WS-OPSCAN-COUNT	Comp-3..sign	3
WS-STUDENT-COUNT	Comp-3..sign	3
WS-RESP	Numeric	3
RECORD-COUNT	Numeric	4
WS-ERROR-FLAG	Alphanumeric	3
YES-ERROR	Condition	
NO-ERROR	Condition	
WS-HEADER-FLAG	Alphanumeric	3
IT-IS-HEADER	Condition	
NOT-A-HEADER	Condition	
AV-PROG-NUM	Alphanumeric	6
AV-MESSAGE	Alphanumeric	70
HOLD-TRUNCATION	Numeric	4

SOME EXAMPLES OF THE ISOLATED DEAD DATA

```
                        60 lines unshown
 61   05  WS-OPSCAN-COUNT          PIC  S9(4)     <= = = Defn
                         2 lines unshown
 64   05  WS-STUDENT-COUNT         PIC  S9(4)     <= = = Defn
                         2 lines unshown
 67   05  WS-CLASS-COUNT           PIC  S9(4)     <= = = Defn
                         5 lines unshown
 73   05  WS-RESP-COUNT              OCCURS 14 TIMES.  <= = = Defn
 74     10  WS-RESP                PIC 9(3).      <= = = Defn
                        16 lines unshown
 91   05  RECORD-COUNT             PIC 9(4)       <= = = Defn
                        52 lines unshown
144   05  WS-ERROR-FLAG            PIC  XXX.      <= = = Defn
145                    VALUE 'NO '.               EVA01450
146     88  YES-ERROR               VALUE 'YES'.   <= = = Defn
```

FIGURE 9.
NON REFERENCED PROCEDURES

```
                        541 lines unshown
542 100-MAINLINE SECTION.                  <= = = Defn
543 110-PROG-SETUP-CONTROL.                          <= = = Defn
                         34 lines unshown
578 120-REPORT-AND-FILE-CONTROL.                     <= = = Defn
                         22 lines unshown
601 130-CLOSE-AND-STOP.                              <= = = Defn
                         74 lines unshown
676 1510-PASS-INPUT-DATA.                            <= = = Defn
                         53 lines unshown
730 1999-EXIT.                             <= = = Defn
                          3 lines unshown
734 2010-BEGIN.                            <= = = Defn
                          5 lines unshown
740 2050-START.                            <= = = Defn
                        268 lines unshown
```

Functional Enhancement. Exercises can be set up to change the logic in an existing routine, add another column to a report, modify or add a new summary to a report, or modify an input or output file. The student would be required to do the analysis using the tools, show how the tools were used, and present the new design. The changes could then be incorporated and tested using the test tools. Most students get very little exposure to setting up a test management system using testing tools.

SUMMARY

Exposing students to CASE for existing systems will give them some introduction to software maintenance, and an introduction to reengineering CASE technology. This technology will be prevalent in the 90's, and will give students some experience in making the transition into the next generation of software development. Much of the 70 billion lines of existing COBOL code will gradually be phased into new systems through re-engineering tools. These tools will include reverse engineering repository based tools, repository based COBOL generators, reusable code managers, and COBOL object oriented support. Tools available today such as the Micro Focus workbench, Viasoft's VIA Renaissance, and Intersolv's Design Recovery can do a good job of introducing some of these ideas.

REFERENCES

Bush, Eric, "CASE for Existing Systems", Information Strategy, Spring, 1991.

Eliot, Lance B., "Do Object-Oriented Techniques and COBOL Mix?, Database Programming and Design, Jan. 1992.

Gunn, Mary T. "Software Re-engineering- A Case Study and Lessons Learned", National Institute of Standards and Technology Report, 1991.

Maxson, Emerson, Improved Maintenance Techniques: The Impact of Software Maintenance Tools on the Application Portfolio:, ISECON 90 Proceedings, 1990.

McMullen, John, "CASE Tackles Software Maintenance", Datamation, Jan. 1, 1991.

Snell, Ned. "Using CASE to Rebuild Software", Datamation, Aug. 1, 1991.

Ridgeway, Michael, "Curriculum Shortfall", Datamation, Dec. 15, 1988.

Stair, Ralph M. Jr., "Using Application Redesign to Maintain Continuity and Profitability", Journal of Systems Management, Aug. 1991.

COOPERATIVE LEARNING IN THE IS CURRICULUM

Susan M. Moncada
Indiana State University

ABSTRACT

In addition to being technically competent, information systems professionals need to possess effective communication skills. Cooperative Learning is an alternative teaching strategy that capitalizes on group work and fosters the development of interpersonal communication effectiveness skills. Cooperative Learning is different from the other forms of group oriented learning? In particular, it is characterized by positive interdependence and individual accountability. Examples of true Cooperative Learning methods include Student Teams Achievement Divisions (STAD), Team Games Tournaments (TGT), and Jigsaw II. This presentation provides an overview of Cooperative Learning theory and suggests ways to implement Cooperative Learning activities into the CIS curriculum.

IMPLEMENTATION ISSUES OF SOFTWARE FAULT TOLERANT SYSTEMS WITH CASE TOOL

Sung Yun Shin, Ali R. Salhenia, Bin Cong
Computer Science Dept.
South Dakota State Univ.
Brookings, SD 57007

Rex E. Gantenbein
Computer Sciecne Dept.
Univ. of Wyoming
Laramie, WY 82071

Abstract

Fault tolerant software can make a valuable contribution in the application of computers to critical tasks. The most likely uses of fault tolerant software are in fields that demand very high reliability. A CASE tool provides many features that improve software productivity and reliability. Using CASE tools in Software Fault Tolerant (SFT) system development can increase the reliability of SFT techniques. In this paper, we describe the common features of well known SFT systems such as Recovery Block, N-version Programming, Consensus Recovery Block, and Recoverable N-version Block. Also, we investigate the important features and implementation issues of CASE tools, and the effect of using such tools on SFT systems in each phase of the software development lifecycle.

1. INTRODUCTION.

An engineering approach to software development using CASE tools enables the developer to produce software on time, within budget, and in accordance with user requirements.

One important aspect of these requirements concerns the reliability of the software. The use of computers for life-critical systems demands extremely high reliability of the computing functions as a whole. Applications that pace the state of art in this regard include spacecraft, fly-by-wire systems for passenger aircraft, safety systems for nuclear reactor, and traffic control system for tracked vehicles and aircraft [Hecht]. The need for high reliability of software components in these life-critical systems has become more apparent with the increasing functionality being ascribed to the software. One result of this increased functionality is the recommendation of various and diverse designs for achieving fault tolerant system.

CASE tools are rapidly becoming more common tools in software development. A CASE tool would obviously be important tool for SFT systems development, because they substantially reduce or eliminate many of the design and development problems inherent in the medium of large SFTs.

Another goal of CASE technology is to separate the application program's design from the program's code implementation [Evans]. Generally the more independent the design process is from the actual code of the module, the better. However, this condition would be the necessity in SFT system design since almost all SFT structures are based on redundancy of software components. "Independent design" should be taken here to mean that programming efforts are carried out by individuals or groups that do not interact with each other with respect to the programming process. Whenever possible, different algorithms, programming layouts, environments, and CASE tools can be used in each separate effort to minimize the probability of similar errors in the redundant versions [Avizienis].

In this paper, we discuss the common features of well known fault tolerance techniques such as Recovery Block (RB) [Randell], N-version programming (NPV) [Avizienis], Consensus Recovery Block (CRB) [Scott], and Recoverable N-version Block (RNB) [Shin]. We investigate the special features and notions of CASE tools for SFT system development for each phase of the software lifecycle (using waterfall model). We discuss why development of the SFT systems should not be the same as ordinary (non-fault-tolerant system) software systems and the important issue of SFT systems implementation with CASE tools.

2. SOFTWARE FAULT TOLERANCE TECHNIQUES (SFT)

Within the discipline of SFT, three distinct approaches have emerged. These approaches are the RB, NVP, and combined approaches such as CRB or RNB.

2.1 RECOVERY BLOCK (RB)

The RB scheme for software faults is based on a time-honored hardware approach to fault tolerance, that is, switching to spare components when an error is detected. In this approach multiple versions of a block of code coexist; if one produces an error, another is tried in the hope of producing a correct result.

2.2 N-VERSION PROGRAMMING (NVP)

NVP, like the RB scheme, relies on software redundancy to provide fault tolerance. The basis for the redundancy is the broadcasting of inputs to multiple logical units, with the multiple outputs sent to an adjudicator. This component performs error detection and propagates a result.

2.3 COMBINED APPROACHES TO SFT

Although RB and NVP are similar in a distributed environment, each does have features that are suited for (or, rather, that avoid the limitations of the other in) particular applications. Since the two are not incompatible, we can easily capture the strengths of both in a single construct [Gantenbein].

The CRB utilizes features of both RB and NVP. The method requires independent development of N >= 2 component versions, an acceptance test and an adjudicator. The boundaries of the CRB are established by selecting logical checkpoints. At these locations (as in NVP) the results of the N versions are compared by a decision algorithm. If two or more of the versions agree, the output is deemed correct and is propagated out of the block.

The RNB, a RB consisting of a primary and alternate blocks and an acceptance test, is nested inside each of N >= 2 independently designed versions. Each version receives the same inputs and processes them concurrently. Within each version, the primary block initially computes a result, which is submitted to the version's acceptance test. If this result is accepted, it is sent to an adjudicator, which synchronizes the results from all versions and applies a decision algorithm to choose a result to be propagated from the block.

3. USING CASE FOR SFT SYSTEMS DEVELOPMENT.

3.1 CASE Tools For the Ordinary Software Systems Development

For many software development organizations, design and development times will almost always be reduced by using CASE tools, but significant benefit of CASE tools is not just time [McClure]. Their most satisfying benefit comes in the form of insurance that the job is being done properly to the user's specification. CASE tools yield a tremendous benefit in revealing many requirements, from before implementation begins to the end of the development [Fisher]. However, we should realize that much of the actual value received from CASE tools largely depends on how well they are integrated into the software development process as well as the kind of software project being developed.

3.2 USING CASE TOOLS FOR SFT SYSTEM DEVELOPMENT.

In the previous sections, we briefly described typical SFT systems and how CASE tools are used for ordinary software system development. As we discussed, almost all SFT systems are based on software redundancy. They require independently designed, redundant components. However, the problem of common software faults giving rise to similar errors in redundant software still remains open. It may be that careful analysis of a system design and requirements can identify potentially dangerous or difficult components of a software for special care during later phases of development and testing.

We now discuss the use of CASE tools in SFT systems development to provide new opportunities to use analysis techniques the software lifecycle (using the waterfall model).

3.2.1 PROJECT PLANNING

Before we begin to develop an SFT systems, we must have some idea of the activities involved, who will do them, how long they will take, and how much they will cost [Fisher]. The project schedule thus breaks down into a series of tasks. A CASE tool can assist the developer in planning and laying out tasks and schedules according to the Work Breakdown Structure (WBS) and work package descriptions. The use of CASE tools for SFT systems development is pretty much the same as for ordinary software systems. However, we must consider the redundant modules and extra SFT components such as the acceptance test in the RB, the Adjudicator in NVP, the acceptance test and adjudicator in

both CRB and the RNB. CASE tools can provide the means for creating a WBS for the SFT systems. Although the WBS of SFT systems should be different that the WBS of ordinary software systems, most CASE tools can be used for SFT systems with a little modification of the symbols used in the WBS. The modification would only be needed for the SFT parts of the software system; we usually use SFT techniques only on the critical parts of the system, since the implementation of SFT techniques is very expensive in terms of development and resources.

Figure 3-1 shows how special symbols (double line boxes) can be used to represent SFT components in a system. These symbols represent the redundant components and additional needs in the SFT. They represent the critical part of a chemical distillation system that we designed with Excelerator. CASE tools used for SFT systems should be able to represent the SFT parts of the system so that these SFT structures can be isolated.

In order to use CASE tools to develop a work package for SFT systems, they should be able to perform activities such as defining system response time in an SFT structure, specifying a piece of hardware to support SFT systems, developing a model to estimate resources for SFT systems, or computing reliability of an SFT systems. A graphic illustration of events in the lifecycle, such as a PERT chart, is also needed to distinguish SFT components in the entire system. Scheduling, planning, budgets, and measurement must be provided as the basis for planning and monitoring the development of the SFT system.

3.2.2 REQUIREMENT ANALYSIS

A requirement analysis tool allows the systems engineer to initiate, design, complete, modify, and maintain SFT systems. CASE tools can also provide special features for requirement analysis and documentation of SFT systems.

Each requirement in the requirements specification should reflect the features and constraints of each component and the whole structure of an SFT system, as well as the SFT techniques such as RB, NVP, CRB, or RNB. CASE tools should be able to define the methods for recovery (backward recovery or forward recovery) and response time in an SFT systems.

Requirement traceability is an important method of demonstrating the SFT of the structures produced (or the reliability of the product) from user requirements. Usually, this is demonstrated in steps by showing that the product of the current development step can be traced back to the previous step of the software lifecycle. For example, we should be able to trace back to the requirement specification of SFT structure from the design specification, or from the source code to the design specification. SFT systems traceability at the code-back-to-design step is shown by automatically creating a structure chart from the source and comparing this chart to the structure chart, created during the design phase. Therefore, the requirement specification should show the structure of the SFT system, the components of the SFT approach, and the constraints of the SFT technique used in the project.

3.2.3 SYSTEM DESIGN

Good software is organized as a set of independent modules, each of which can be designed and tested separately. Each module views the others as black boxes with well-defined sets of inputs and outputs. Each module is accessed only through these inputs and outputs [Evans]. This logical segregation of functionality is not only a key factor of ordinary software development but also a main factor of SFT system implementation. The success of developing SFT systems totally depends on the independently designed versions of the components in the NVP, the alternate versions of RB, CRB and RNB approaches. CASE tools used in SFT systems must support the functional design of the SFT system. In this functional design, we should not describe the possible code or precise logical pseudo code for each component of SFT system, since the variety of programming in each component of the SFT system is a crucial factor. We must be very careful to use CASE tools for the SFT system design phase. If CASE tools make too detailed the logic of the SFT components to the programmer, it could cause make a common error among the redundant components of SFT structure. We should set the functional design boundaries (design guideline) for the SFT structure design, but the design specification should only delineate how the input elements in each component of the SFT. structure are transformed into the outgoing data elements. CASE tools support decision tables, decision trees, and detailed specification, which are very good techniques for SFT system

design. Detailed pseudo code, however may eliminate the independence of implementation in SFT components.

CASE tool supported data flow diagramming is an appropriate design tool that shows all of the input and output data flow entering and exiting the system. There are no other sources and sinks hidden in lower-level data flow diagrams. The topmost data flow diagram permits the entire underlying structure to be treated as a black box. This important design concept provides a powerful structuring mechanism that allows a separation or segregation of functionality.

CASE tools for SFT systems should support the creation of the topmost data flow diagram that represent not only the data flow, but the SFT techniques to used. The design structure of the SFT systems should also show reconfiguration scheme, recovery methods, and error-detection method. Examples of the topmost data flow diagram are presented in Figure 3-2 through Figure 3-5.

The design of the chemical distillation control system shows the structure of the RB in the dotted box (Figure 3-2). The double circled bubbles show the components of the RB (primary block, alternative block, acceptance test). The mechanism of the RB would be the same as described in the previous chapter. Note that the reconfiguration of the system must be specified in the software specification document.

Figure 3-3 shows the NVP structure in the dotted box. The double circled bubbles show the components of the approach (N modules and adjudicator). This data flow diagram shows the error detection and recovery mechanisms in the dotted box.

Figure 3-4 and Figure 3-5 show only the critical parts of the SFT system that use the combined SFT techniques such as CRB and RNB approach. They also show the structure and recovery mechanism of the combined approaches.

The CASE tools used in SFT systems should be able to draw data flow diagram for SFT systems. The system design documentation must include the design constraints, reconfiguration techniques, error detection mechanisms and recovery mechanisms. In the design phase, a CASE tool (Excelerator) is obviously very helpful. However, we must take special care to avoid dependency in the logic design, much like precise pseudo code problem.

4. CONCLUSION

The ultimate goal of SFT is to increase the reliability, availability, and safety of critical applications. The ultimate goal of a CASE tool technology is to separate the software system design from the implementation of program code. As we discussed, CASE tools can be very important for SFT systems development. While the classic approaches to software SFT do increase the complexity of the large software systems, some fundamental problems limit their effectiveness. The use of CASE tools for SFT systems described in this paper address some of these problems. However, further study is needed to develop a methodology to generate independent designs and code for SFT system components.

No consideration has been given in this paper to the maintenance of SFT systems. The maintenance phase of SFT systems must be considered on both ordinary and SFT systems, but, no CASE tools are available specifically for SFT systems.

The problem of common software faults giving rise to similar errors in redundant software also remains open. Careful analysis of a system design with CASE tools can identify potentially dangerous or difficult components of a system for special care during later phases of development and testing. Techniques have been proposed for performing such analysis [Kemmerer], but they have not been fully explored in terms of fault tolerance on CASE tools.

Finally, none of the features of CASE tools used in SFT systems development makes good provision for avoiding specification and design faults. We plan to investigate several other problems within CASE tools so that they and the reliability of the systems developed with them can be improved.

REFERENCES

[McClure] Carma McClure, "CASE is Software Automation" Prentice Hall 1989. pp. 71-118.

[Shin] Sung Yun Shin, Rex E. Gantenbein, John R. Cowles, "Evaluation of Combined Approaches to Distributed Software-Based Fault Tolerance" Proc. of 1991 Pacific Rim International Symposium on Fault Tolerant Systems. Tokyo, Sep. 1991, pp

70-75.

[Gantenbein] Rex E. Gantenbein, Sung Yun Shin, Z. Wang, "Software Fault Tolerance in a Distributed Real-Time Control System" Proc. of 4th ISMM/IASTED International Conf. on Parallel and Distributed Computing and Systems, Washington D.C. Oct., 1991 pp 61-64.

[Fisher] Alan S. Fisher, "CASE Using Software Development Tools" Willy, 1988, pp. 23-119.

[Evans] Michael Evans "The Software Factory", Willy, 1989 pp. 107-139.

[Horning] J.J. Horning, "A Program Structure for Error Detection and Recovery", in Lecture Notes in Comp.Sci., Vol. 16 New York:Springs-Verlag,1974, pp 172-187.

[Hecht] H. Hecht, "Fault-Tolerant Software for Real-Time Applications." Compt. Surveys, 8,4, 1976, pp 391-407.

[Randell] B. Randell, "System Structure for Software Fault Tolerance", IEEE Trans. Software Eng., Vol. SE-1 pp. 220-232, June 1975.

[Avizienis] A. Avizienis, "The N-version approach to Fault- Tolerance Software", IEEE Trans. Software Eng., Vol SE-11 Dec. 1985, pp 1491-1496.

[Scott] R.K. Scott, J.W. Gault, and D.F. McAllister, "The Consensus Recovery Block" in Proc. Total Systems Reliability Synp., 1983, pp 74-85.

[Kemmerer] R.A. Kemmerer, "Testing Formal Specifications to Detect Design Errors", IEEE Trans. Software Eng., Vol SE-11, Dec. 1985, pp 1502-1510.

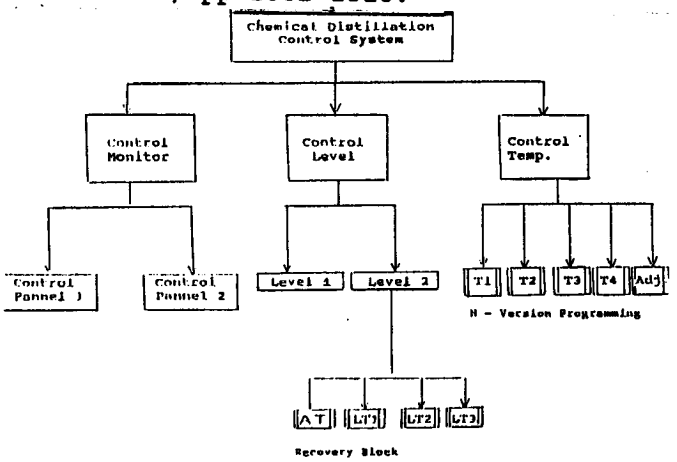


Figure 3-1 Work Breakdown Structure of SFT system

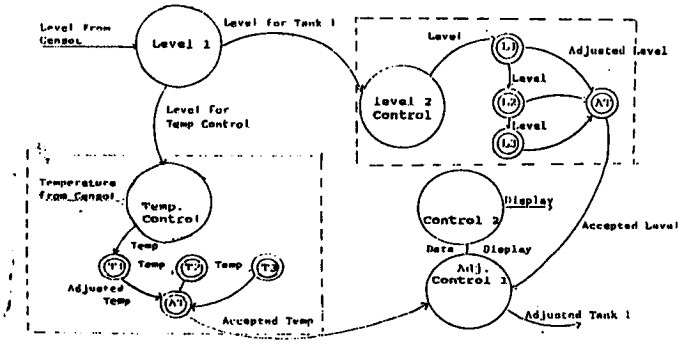


Figure 3-2 Recovery Block Data Flow Diagram

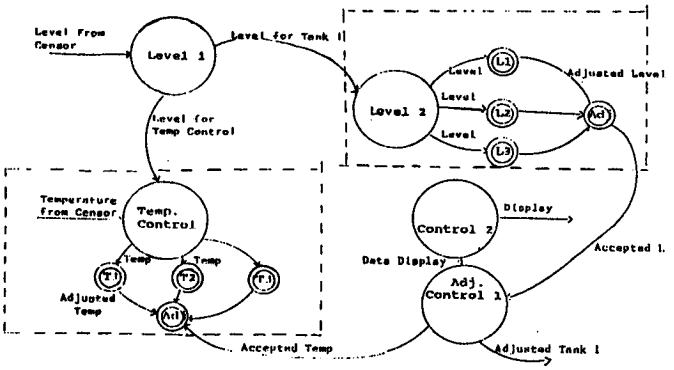


Figure 3-3 N-Version Data Flow Diagram

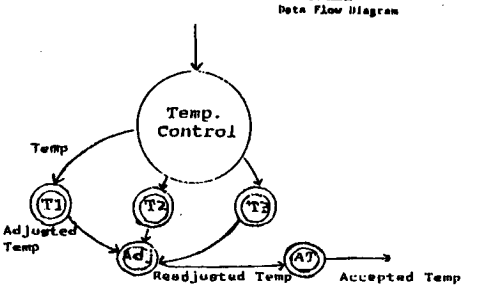


Figure 3-4 Consensus recovery Block Data Flow Diagram

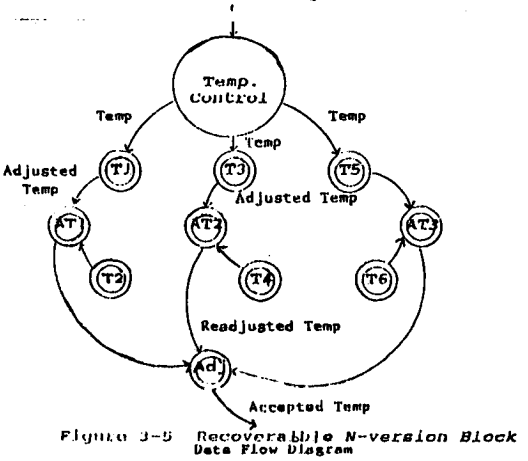


Figure 3-5 Recoverable N-version Block Data Flow Diagram

END USER APPLICATION DEVELOPMENT

AND PROBLEM SOLVING IN AN INTRODUCTORY COMPUTER COURSE

IS CURRICULA TRACK

by

Craig A. VanLengen, Ed.D.
College of Business Administration
Northern Arizona University
Box 15066
Flagstaff, AZ 86011-5066
(602) 523-7392
vanlenge@nauvax.ucc.nau.edu

and

James A. O'Brien, Ph.D.
College of Business Administration
Northern Arizona University
Box 15066
Flagstaff, AZ 86011-5066
(602) 523-7381
jao@nauvax.ucc.nau.edu

ABSTRACT

A system development life cycle for end user development is presented along with alternative means for end users to develop their own computer solutions. The solutions range from the end user executing existing menu commands and using predefined report formats to the end user writing and executing programs using a DBMS procedural language.

INTRODUCTION

Introduction to computers courses at many colleges and universities still include instruction in computer programming. The objective of such instruction is to teach students problem solving, an appreciation of the process that information system (IS) professionals follow to develop computer systems, and a deeper understanding of how a computer works. Since the vast majority (over 95% at many schools) are non-computer majors, teaching computer programming as a means of problem solving is open to question. This practice may be a waste of computer lab resources and limited course time. This paper suggests that it may be more useful to teach students end user application development and problem solving throughout the course.

END USER DEVELOPMENT CYCLE

This paper suggests that a modified problem solving and systems development life cycle for end user development can be taught in an introductory computer course. This life cycle might have the following steps or phases:

a. Problem/needs understanding and definition

b. Application of creative thinking to suggest alternatives.
c. Application of critical thinking to evaluate alternatives.
d. Select an alternative.
e. Construct/implement solution.

Problem/Needs Understanding and Definition

Like traditional problem solving, the specific problem and its cause should be identified. Since users are conducting their own analysis communication problems are avoided. Users determine and document what information is needed. Objectives of the needed information are identified and documented. Problem decomposition might be used to break the problem down into smaller units that are more manageable. Extraneous data and information are excluded or ignored to assist in solving the problem identified. Output and inputs are identified, along with the form they must be in for processing. Algorithms and data needed are specified for outputs that must be created. In most cases the inputs are data that is available in existing organization databases.

Application of Creative Thinking to Suggest Alternatives

In this stage, users attempt to visualize how the system could work or what would they like it to produce. Brainstorming with other users and suggestions from computer professionals can be used to generate alternative means of solving the problem. It is important at this stage to remain flexible and to consider different approaches from the traditional, so that the means of solving the problem are as broad as possible. It is important to use knowledge of solutions gained from previous development efforts. Patterns or similarities with prior problems should be investigated to suggest solutions. End users should also analyze the different tools they have available and know how to use to help solve the problem. For example, if data is to be accessed from a single database table and manipulated with mathematical formulas, using a spreadsheet may be the best way of solving the problem. However, if data from multiple database tables are to be accessed, extracted, and combined, a database management system might be the best approach.

Application of Critical Thinking to Evaluate Alternatives

The solution to a problem must be matched to the goals and objectives of the end user department and organization. Thus solution alternatives should be evaluated based on the goals and objectives and the computer hardware, software and personnel resources that are available. User proficiency with specific hardware and software tools should be given a high priority. Then alternatives should be compared against each other and ranked.

Select an Alternative

The highest ranked alternative that users are proficient in should normally be chosen. A large difference may sometimes exist between the alternative that involves tools that users are proficient in, and an alternative that consumes less resources. Then end user training should be considered so users can obtain proficiency in the tools required for implementation of the solution that requires less resources.

Construct/Implement Solution

The end user will create the solution using the capabilities of the software tool chosen. The solution may be implemented in one of the four methods:

a. Using menus and predefined reporting formats.
b. Using menus and designing their own reports.
c. Using a command interface.
d. Writing and executing programs (writing procedure or command files).

End users should develop and test their solution until it is evaluated to be correct. It is important that end user developed software be tested and evaluated by someone other than its end user developer. After the program is completed, it can be implemented by the end users and used in performance of their job functions.

Examples of End User Development

At Northern Arizona University different types of problems and tools are being used to teach end user problem solving and development. Here are some examples that have been used with a database management system as the solution tool.

Example 1: Using the menu and predefined reporting formats.

A small specialty retail store is having problems in customer awareness of its goods and services. Revenues are not sufficient to cover the costs of keeping the business open, much less provide a profit for the owner. The owner has very limited funds available for advertising. Even if the organization could afford advertising in television, radio, and newspapers there would be a question of the effectiveness of the coverage.

Problem: By examining the current situation the owner/manager of the business realizes that revenue must be increased. The lack of revenue, or the cause of the problem is attributed to lack of consumer information about the goods and services provided by the organization.

Alternatives: Advertising and promotion is considered to include television, radio, newspapers, and direct mail.

Evaluation: Television, radio, and newspapers are expensive and also would not specifically target the consumers that the owner wants to reach for a specific promotion. Brochures can be created listing the goods and services of the company and the specific promotion. A microcomputer database is available with the names, addresses (street, city, state, and zipcode), and telephone number. The owner is somewhat comfortable with the menus and is aware of the built in mailing list features available with the database management system.

Selection: The direct mail alternative is selected and the database is acquired and loaded on the computer system of the owner.

Construct/implement: The end user (business owner) would be classified as being capable of using the menu and predefined reporting formats. The solution would be to use the menu system to create the labels.

Details of the life cycle steps or phases for examples 2-4 will be presented at the conference.

Example 2: Using the menu and designing their own reports.

In this example the owner would like to have a report of the entire database alphabetic within each zip code area. The database could be arranged in this order with a sort command or an index could be created. The user wants to view the database in many different ways. Rearranging the database each time is time consuming and unnecessary. Separate index files can be created to provide the user with the capability of viewing the database in different ways.

In this case the index solution is selected menu options are used to create an index to the database. Then menu options are selected to access the report generator and to define the report format. The report format will be named. Again from the menu, the user selects the appropriate option to retrieve data and provide the name of the report format for output appearance.

Example 3: Using the command interface.

In this example the user has incomplete information of the data values that she wants to search the database for. The user wants a means of searching, selecting, and reporting when she only knows part of the data value of two different fields. The substring function allows this capability and can be executed from the command prompt.

Commands are entered to: make the database available for use, use the previously defined report format, to search the database for the two conditions.

Example 4: Writing and executing programs (writing procedure, program, or command files).

The owner would like to convert from manual calculation of payroll to a computer assisted system. Payroll programs and writing the program in the database management system procedural language are evaluated. The owner decides to create a simple program in the procedural language of the database management system. This option is based on cost and a desire to better understand programming.

Algorithms based on the manual process for the calculation of gross pay, withholdings and net pay are developed.

The owner creates an employee database file and enters employee data in the database. From the command prompt the program editor is accessed. The algorithm along with other required statements are entered. When the program is complete it can be executed. If syntax errors are found the user returns to the program editor to correct the errors. Prior payroll data is used to test the program. Testing should continue until the program results agree with the previously computed payroll output. The program should be tested by other end users to ensure that it is working correctly prior to using it in their job.

CONCLUSION

There are many different ways that end users can develop computer based solutions. The end user solution can be as simple as using menu commands and default report formats or as complicated as programming in a procedural language.

An end user life cycle has been presented along with different means that are available to the end user for development of a computer based solution. This end user approach to development is currently being used and evaluated at Northern Arizona University.

BIBLIOGRAPHY

Cook, R. (1990). Full circle. Byte, 15(8), 211-214.

Dekleva, S. M. (1991). Introduction to programming using dBASE. Journal of Information Systems Education, 3(2), 10-15.

McMullen, J. (1989). End users create PC applications. Datamation, 35(23), 41-43.

Obermeier, K. K. (1990). Natural selection. Byte, 15(8), 217-222.

Schocken, S. (1992). The art of business programming with dBASE III plus and IV. Santa Cruz: Mitchell McGraw-Hill.

New Opportunities for IS Educators: Client-Server Database Systems

David Munro
George Sargent
Robert Horton
University of Wisconsin - Whitewater

ABSTRACT

Downsizing applications from minicomputers or mainframes is a current trend that is being discussed daily in the computer media. The primary force driving downsizing is maturing of client-server database systems. Because of downsizing, MIS educators have new opportunities in three areas: (1) development of new curriculum content, (2) development of new delivery methods, and (3) expanded student clientele.

The curriculum is impacted by client-server database systems in allowing for and further supporting distributed database concepts, networking, and event-driven programming. In addition Object-Oriented Programming Systems (OOPS) are philosophically compatible with client-server systems. The client-server approach allows students to use multiple front-ends for the same database server. This helps students to understand the differences between database concepts and database products. In additional the client-server approach will allow the integration of graphical user interfaces and database material in the same course.

A Framework for New Directions in Information Systems Education: Breaking old IDOLS

Eli Boyd Cohen
College of Business
Eastern New Mexico University
Portales, New Mexico 88130
(505) 562-2066
email: eli@bradley.edu

There is not a more unhappy being than a superannuated idol.

Joseph Addison
[The Spectator, no. 1 [March 1,1711]]

Abstract

The old paradigm for IS education, as described in the paper, fails to provide students with the knowledge and skills that they need to succeed. This paper examines the paradigm shift in Information Systems (IS) education and presents it in the form of the word framework IDOLS for Integration, Distinction, Orientation, Logical Teaching Practices, and Study of Current Practices. This framework should be useful in course and curricular development.

Introduction

Joseph Addison's quote about outdated idols pertains to the idols of IS education as well as to those of other theologies. Environmental changes force us to abandon our old, inadequate idols in search of new idols or paradigms. Paradigms are the ways science understands phenomena and, as Thomas Kuhn points out, old paradigms die hard.(10) Long after they have ceased to be useful, their adherents continue to promote them. The same is true in IS Education. Our paradigms are failing, and reformers in IS education, from text book authors to curriculum developers, are promoting new ways of teaching IS. The following is a review of the developments these leaders advance.

This article organizes IS educational reforms into five categories using the acronym IDOLS, as seen in Table 1.

"I" is for Integration
"D" is for Distinction
"O" is for Orientation
"L" is for Logical Teaching Practices
"S" is for Study of Current IS Practices

Table 1. Reforms in IS Education:
The IDOLS Acronym

Using this acronym as a tool, this paper consolidates the critiques of various leaders in IS education into a set of reforms. This work is dedicated to those pioneers who see the inadequacy of our old paradigms for IS education and courageously break new ground.

The Old Paradigm

This paper asserts that the leaders in our field are breaking with the old paradigm of how to teach IS. What, then, is the old way?

Here are some of the characteristics of the old idols for how to teach IS:

1. As new subject matter is determined to be important to IS, we will add it to the curriculum. We do not remove old material as it forever retains its importance.
2. Subjects that are of critical importance to all IS majors are taught in core courses. Each subject is taught in its own course.
3. The introductory IS course should give business majors a flavor for what IS professionals do for a living.
4. We are "the computer major". Courses involving computers should be offered through our major.
5. Our graduates will be hired as programmers, and because of their superb education, be on the fast track to become analysts.
6. Our graduates will develop large systems on large computers.
7. The best background for teaching IS is to be a specialist in one of the core areas. There is no need to understand another's area in depth.

This list gives a flavor for the old idols that have served us well in the past. As those who have taught IS for over a decade can attest, this is no strawman. These characteristics give a realistic view of how we perceived our mission and our profession. Now, let us examine, using the "IDOLS" acronym, what changes in IS education our leaders recommend.

The IDOLS Framework for Exploring the Developing Paradigm

The IDOLS framework has five components. It suggests that we should examine changes or needed changes in Integrations, Distinction of the discipline, Orientation of the discipline, Logical teaching practices, and the Study of current IS practices. Let us now examine each of these in turn.

Integration

The 1991 DPMA Model Curriculum for CIS Education speaks of courses entitled "Systems Development I" and "Systems Development II" in place of courses entitled "Data Base" and "Systems Analysis" found in earlier curriculum models.(11) This coordination and integration of content from related courses is an example of the integration in IS education; the content of each course is intertwined with that of the others. The courses build on each other.

Heinrichs and Williams emphasize that an integrated curriculum, such as the one that they propose, will provide students "with a greater understanding of the interrelationships" among the systems that comprise IS.(8)

Integration is also evident on the level of the profession itself. The Institute for Certification of Computing Professionals (ICCP) sponsors the Certified Data Processor, Certified Computer Programmer, and Certified Systems Professional certificates. In the past, the ICCP administered different examinations for these different certificates. Currently, passing a common core examination (on general IS topics) is required for each ICCP certificate. Separate specialty examinations differentiate the three areas of certification.

At first blush, integration of courses is nothing new. For years, colleagues have informally collaborated to improve two or more courses by integrating them. For example, Larry Cornwell, teaching database, and this author, teaching systems analysis, had the students develop analyses for a common case. What is new is the formal integration of content across courses.

EXISTING TOPICS. One of the potential problems with any curriculum is ensuring that important topics have a home course. The term home course is used here to mean the course where the topic is first introduced.

An example of this problem in the IS curriculum is finding a home course for the topic "elements of data structures". Should we expose our students to this topic first as part of COBOL, in database, or perhaps in the introductory IS course? Each of these potential home courses poses its own set of advantages and disadvantages. The issue is not so much which course is chosen as the home course, but that one is chosen.

If a topic, such as data structures, does not have a home, it is taught in any given course at the whim of whichever instructor teaches that course each year. Some years the topic may be repeated in a variety of courses while in other years students may completely miss coverage of data structure in their course work.

Integrating the courses in the curriculum solves these problems. All topics are assigned a home course. This does not preclude covering the topic in a later course as the need arises. The student in such an integrated curriculum may be exposed to data structures, for example, once, twice, or more. In this case, the re-visiting of a topic is by design, each time at a deeper level. Such an approach to curriculum development is known as the spiral approach.(7)

DPMA MODEL CURRICULUM. The DPMA Model Curriculum follows such a model. The contents of the DPMA Model Curriculum's proposed courses are integrated in a spiral curriculum so that students are challenged by the important topics again and again as they progress through school, each exposure at a deeper level of understanding.

The model curriculum first identifies clusters of skills that our graduates need and progresses from this base to establish proposed courses. Some of these courses address the needed skills in only the lowest of terms, using Bloom's taxonomy of educational objectives.(4) Later courses build on this knowledge, re-visiting the topic and refining the students' understanding of it.

NEW TOPICS. This approach also solves the second problem in developing IS curriculum: schools cannot offer new courses to embody each and every important development in our field. Expert systems, group decision-making, and international issues would each deserve a separate course under the old curriculum model. The spiral model allows the curriculum to remain responsive to changing technologies and interests. For example, consider the important issues of ethics and internationalization of the curriculum. Rather than just adding new courses in each of these two areas, the new DPMA Model Curriculum will incorporate these issues again and again in various courses. Likewise database concepts will be taught and reviewed in the Introductory Course, Systems Development I and Systems Development II.(13)

Distinction

Distinction refers to making the IS curriculum different from the other curricula. Some twenty years ago, faculty accepted as a given that courses dealing with "computerization" and training students to become business programmers and analysts belonged to the IS program. In recent years, however, many campuses have discovered other departments offering classes and even majors that directly compete. It is not uncommon to find com-

puter science and mathematics departments responded to their shrinking enrollments by offering IS, even if IS is taught elsewhere on campus.[1]

In recent years, we find even functional areas offering their own information systems courses. We find Accounting teaching Accounting Information Systems, Marketing teaching Marketing Information Systems.

These politics, for better or worse, are not new. This situation parallels the distribution of statistics classes across disciplines, for example. Some IS professors will fight for the "computer applications" territory, but this author asserts we have already lost the battle. We do not "own" information systems any more than the Statistics Department owns statistics for nurses or Computer Science owns computer art.

Thinking that equates IS with "computer classes" leads nowhere. For many, only a blurry line separates IS from the other disciplines. To become a distinct discipline, we need to demonstrate that which makes IS distinct. We need to clarify our discipline to students, colleagues, and administrators alike. The problem is, do ***we*** know what IS is?

DEFINING IS. One way to define a field involves analyzing the contents of its introductory texts. Such an analysis for IS leads to a quandary. Our introductory text book authors have not settled on a single set of accepted topics. For example, McLeod's Introduction to Information Systems draws heavily from the field of management, while Ahituv and Neumann also draw from psychology and mathematical fields in their definitions of the field.(14, 1) While the diversity is not inherently bad, it does blur the distinction between IS and its constituent fields. This diversity reflects how IS is viewed by our own professors; different schools have different emphases.(6)

My own view is that IS is in the borrowing stage of its development, borrowing principles and research from more established fields. The next developmental stage awaits the genius who will develop the models IS needs for self-definition. In the meantime, we need to emphasize our focus on information, not on computers. Our discipline is the one that provides management with information, for use in decision making and for use as a product. To accomplish our goal, we need to understand what decisions managers make, how they make them, and how to value information. We also need to be conversant in the use of technology so far as technology promotes our goal.

Orientation

Distinction, described above, refers to how we are different from other fields. Orientation refers to where we believe our graduates are headed. The differing and changing needs within the field of IS suggests that we need to address various orientations within IS.

CAREER-PATH ORIENTATION. One approach to IS education is to provide options within our programs for those who wish to pursue various careers paths. For example, the Association for Computing Machinery (ACM) Information Systems Model Curriculum provides for different strands or foci within the curriculum, mostly along the lines of the potential occupations, such as programmer and analyst.(16) These strands are offered because currently these are the job opportunities.

We need to recognize and train for other jobs of the future. As IS educators, we are challenged to predict what future occupational skills will be needed by our graduates and to refine our curriculum accordingly. As we do so, we are aiming at a moving target.

Business's reliance on modern technology, including personal computers, alters the job opportunities our graduates are likely to encounter.(2) We now find our graduates taking on jobs not only in programming, analysis, and consulting, but also working as information center liaisons and expert system engineers. As the work place evolves, so must our programs.

NEED FOR BUSINESS ALLIANCE. A corollary of the above can be stated as follows: as the IS curriculum strives to keep pace with the changing needs of the work place, we increasingly must keep in touch with the work place. To accomplish this, we must know what tools, skills, and knowledge our graduates actually use. One way to obtain this feedback is through strong alumni ties. Another is through advisor boards. In either case, we need an information system for IS education.

Logical Teaching Practices

David Kroenke, author of a popular text for the Introduction to IS course, calls that introductory course the "Widowmaker".(9) Following Drucker's terminology, Kroenke defines a widowmaker as any course that defeats two instructors in a row who in earlier assignments had done well. Kroenke proposes solutions to this problem in his seminars. The solution he proposes is manifest, not surprisingly, in the most recent edition of his text. A good part of the solution Kroenke proposes is to motivate the student by showing how this material is relevant to the student's career.

[1]An extreme example of this occurred at a Midwest university. The duplicate major offered by Computer Science was established by decree of the Provost. That department went on to try to destroy the IS major housed in the college of business. This may be the first reported attack of a computer virus transmitted by a Provost.

Indeed, Hans Anderson speaks of five principles that guide successful teaching: 1) provide rationale for learning, 2) provide reinforcement, 3) give clear objectives, 4) provide practice, and 5) present instruction in sequenced incremental steps.(3) Let us look at the first two of his principles.

RELEVANCE. Philips echoes this solution, suggesting that we must provide learners with evident logic of why this topic is of concern to them.(15) Some topics, such as microcomputer applications, possess face validity as relevant to the students' careers, while others, such as data communications seems foreign and without worth. Our task then is to demonstrate to students why learning the material is worth their while. Kroenke's solution is to provide the student with scenarios in which characters with whom the student can identify save the day and their career through mastery of the topic.

REWARD. A related psychological principle that bears on logical teaching practices is reward. Students find some learning activities, including hands-on computer experience, to be fun and rewarding.

However, some educators advance the proposition that most hands-on computer activity, such as learning a spreadsheet program, belongs to remedial, non-credit courses, not as part of the IS curriculum. The criticism is that there is no enduring educational content to training in a given software package; training is not education.

This author propose that we can overcome the criticism and use the principle of reward at the same time. We should design our courses so that the sought-after hands-on activities act as a vehicle in the teaching of IS theory. The purpose of IS is to solve business problems. Let us organize the material so that students become aware of a problem, analyze it, and then learn to solve that problem using some available tool. After the student can solve this problem, we spiral the curriculum to reveal another facet of the problem or another problem, again to be solved.

In summary, our challenge is to design a logical pedagogy that has the following characteristics:

1) shows students how the topics are be relevant to their careers, and ,
2) uses the popular hands-on software as a vehicle to teach enduring concepts of IS by having students develop solutions to business problems.

Study of IS Current Practices

We are confronted with the curriculum problem that, even now, we cannot cover all the important and relevant material. One solution described above is integration of topics. Another is learning to abandon topics that were important yesterday, but are not important for tomorrow. The orientation reform described above will provide us with feedback from employers, but that alone is not enough.

We must follow our own teachings and discover the critical success factors relevant to our courses and curriculum.(5) Our students need to know some material very well, others only in passing. Longenecker and Feinstein, in spearheading the fine work that led to the 1991 DPMA Model Curriculum, confronted that problem.(12) Their work provides an excellent source as to what topics are perceived as important by members of professional societies. From this work to critical success factors is a small but important step.

Summary

The old idols of IS education are superannuated, obsolete. This paper has tracked the changes that leaders in the field have been making to our pedagogy and curriculum. The acronym "IDOLS" summarizes these improvements in Integration, Distinction, Orientation, Logical Teaching Practices, and Study of Current IS Practices. The leaders call for recreating the curriculum from the ground up so as to position it to meet the needs of today and the future. To paraphrase the biblical character Abram, "It is time to break some old idols around here."[2]

References

1. Ahituv, Niv and Seev Neumann. *Principles of Information Systems for Management (3rd ed).* Wm. C. Brown Publishers, Dubuque, Ia, 1990
2. Amoroso, D.L., J.C. Brancheau, and F. McFadden. "The Senior Executive as Organizational Stakeholder of Microcomputer Technology." *Information Resources Management Journal.* 4(3) Summer, 1991, pp 24-40.
3. Anderson, Hans. *Planning: Rarely Enough But Never Unnecessary.* (unpublished monograph).
4. Bloom, B. S. et. al. *The Taxonomy of Educational Objectives: Classification of Educational Goals Handbook 1: The Cognitive Domain.* McKay Press, New York, 1956.
5. Bullen, C. V and J. F Rockart. "A Primer on Critical Success Factors". *CISR No. 69, Massachusetts Institute of Technology*, June 1981.
6. Denning, E.C., et. al. "Computing as a Discipline." *Communications of the ACM.* 32(1) January, 1989, pp 9-23.

[2]Abram changed his name to Abraham to demonstrate his breaking with the past. Perhaps IS needs to change its name (for example, to information resource management) to demonstrate the breaking of its old idols.

7. Gagne, R. M., L. J. Biggs, and W. W. Wager. *Principles of Instructional Design* (3rd ed), Holt, Rinehart, and Winston, New York, 1988.

8. Heinrichs, L. R. and G. A. Williams. "Recharting the Information Systems Curriculum." *Proceedings of the International Association for Computer Information Systems.* San Francisco, 1991, pp. 274-282.

9. Kroenke, D. "The MIS Course: Remaking the Widowmaker". *Interface*, 10(3), Fall, 1988, pp 2-12.

10. Kuhn, Thomas S. *The Structure of scientific revolution.* University of Chicago Press, Chicago, IL. 1962.

11. Longenecker, H.E., Jr. and D.L. Feinstein (eds). *Information Systems: The DPMA Model Curriculum for a Four Year Undergraduate Degree*, Data Processing Management Association-Education Foundation, Park Ridge, IL, 1991.

12. Longenecker, H.E., Jr. and D.L. Feinstein. "A Comprehensive Survey of USA and Canadian Undergraduate Programs in Information Systems". *Journal of Information Systems Education*, 3(1) Spring, 1991, pp 8-13.

13. Longenecker, H.E., Jr. and D.L. Feinstein. "On Establishing Excellence in IS." *Journal of Information Systems Education*, 3(1) Spring, 1991, pp 26-31.

14. McLeod, R. *Introduction to Information Systems: A Problem Solving Approach.* SRA: Chicago, 1989.

15. Philips, G. "Education and Training for the Adult". *Journal of Systems Management.* April, 1989, pp. 8-9.

16. Tucker, A. B. (ed). "A Summary of the ACM/IEEE Joint Curriculum Task Force Report", *Communications of the ACM.* 34(6), June 1991, pp 68-84.

An Intelligent Kiosk for Student Counseling

Mr. Robert Henry, Sinclair Community College
Dr. Kathryn Neff, Sinclair Community College
Dr. Kenneth Melendez, Center for Artificial Intelligence Applications

I. Introduction

> *They arrive in twos, threes; in business suits, dresses, shorts, mechanic uniforms; they are married couples, single girls with children, bewildered high school seniors with 1.85 GPAs; perplexed parents; auto mechanics who leave greasy footprints on your office carpet and college graduates with Ph.D.s.*

This excerpt from an article by Dennis Rash [1], a counselor at Lakeland Community College, succinctly describes the diversity of the student population at a typical community college.

Increasing numbers of non-traditional, multicultural, and underprepared students are taking advantage of the open-door access to higher education offered at community colleges. While this is certainly a positive trend, the increased diversity and the growing numbers of non-traditional students result in heavy demands on academic counseling resources. While non-traditional students are often highly motivated, most are in need of guidance in establishing academic and personal goals. This increase in demand for counseling comes at a time when institutional budgets are under severe stress; the numbers of professional counselors at most community colleges are holding steady if not decreasing.

The authors of this paper are participants in a project aimed at supplementing their institution's counseling resources by means of an automated advising system. The automated system integrates expert systems technology, database access, and multimedia to produce a "smart kiosk" for academic counseling. Using a touch-sensitive screen, students will engage in a dialogue with the kiosk about courses to take next term, career and major choices, financial matters, the student's academic status and progress, and how many credit hours he or she can handle successfully along with job and family commitments. The "smart kiosk" will consider the student's responses, along with information drawn from the institution's administrative databases, to provide personalized academic advisement. The objective of this automated advising system is to handle routine and predictable questions, so that the human counselors can allocate more time to working pro-actively with the "at risk" population.

The name selected for the system is CWEST (pronounced "quest"), which is an acronym for "Counseling With Expert Systems Technology." While the pronunciation and meaning of "CWEST" might not be immediately apparent, the name is symbolic of our quest for innovative applications of technology. Perhaps more important, the system will also serve as a "quest for success" on the part of the students who will use CWEST for academic counseling.

II. Background

The automated advising project described here originated as an artificial intelligence initiative which was funded by a state grant awarded to the authors' institution in the summer of 1990. As a part of this initiative, a team of academic counselors, faculty members, and computer support staff began work on an expert system for academic advisement. Although the basic objective was to develop and deploy an expert system, the project also served as a faculty development opportunity; faculty team members could gain hands-on experience with artificial intelligence in preparation for teaching classes in AI and Expert Systems.

Academic counseling was selected as the knowledge domain for this project because of the critical shortage of counseling expertise, and also

because advising is an application with high visibility within the institution. Since all faculty, staff, administrators, and students can relate to counseling, it offers opportunities for broad participation in the development of the system. When the system is implemented, it will also provide students with direct, hands-on experience with artificial intelligence.

A number of educational institutions are currently using kiosks to provide read-only access to database information by students (e.g., Genesee Community College in Batavia, New York, and the Maricopa Community Colleges in Phoenix), and at least one institution (Dallas County Community College District) has fielded an expert system for advising that runs in batch mode. The use of multimedia technology for lectures and computer-assisted instruction is also becoming widespread in academia.

Some work is also in progress in the area of integrating these disparate technologies. For example, Ragusa and Orwig [2] describe joint applications research being done by the University of Central Florida and NASA aimed at integrating expert systems and laser-optical devices for information access. The automated advising project described here represents a further advance with the integration of expert systems, database access, and multimedia facilities delivered through a public-access kiosk.

III. System Architecture

In a typical counseling session, an academic advisor will address a number of different questions and concerns of the student. This is especially true in a community college setting, where the majority of the students are adults with jobs and families, and many have returned to school as a result of a personal crisis, such as divorce, loss of a spouse, or loss of a job. Typically the academic, personal, and financial concerns of the student are tightly intertwined, and an academic counselor must consider all of these factors simultaneously. Since these factors are so closely interrelated in the cognitive processes of our experts, it has taken a substantial effort to sort out the separate issues, in order to define the basic structure of the counseling system.

At the same time, some of the functions of the CWEST system have been difficult to define because of the inherent constraints of the computer. Counseling is a human activity, often requiring sensitivity to the student's non-verbal cues, such as facial expressions, body language, and tone of voice. Obviously this sensitivity is beyond the capabilities of the computer. On the other hand, the unemotional nature of the computer can sometimes work as a positive factor. In some counseling situations this neutrality can be used to advantage. For example, some students will probably find it easier to converse with a machine than a human about highly personal or sensitive problems.

The structure that we have developed (Figure 1) represents our consensus regarding the scope and functions that are appropriate for an automated advising system at our institution. The system is modular, with each module addressing a primary concern that applies to almost all students at some point during their academic career.

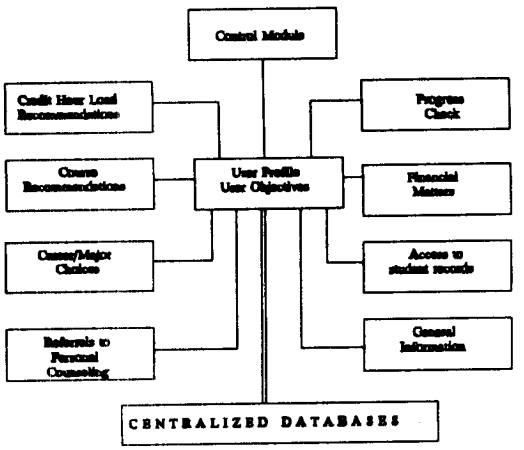


Figure 1. The Logical Structure of CWEST

Control Module

The control module manages the initiation and termination of on-line counseling sessions. When the system is not active, the kiosk will

display two alternating title screens (the "attractor sequence"). Each title screen will have a touch-sensitive area which the user will touch to activate the system.

At all times during a session, the user will have a "quit" option to terminate the session. When "quit" is selected, or if the kiosk remains inactive for a specified amount of time, the system will return to the control module and the attractor sequence will be displayed.

User Profile / User Objectives

The various categories of users (student, counselor, faculty, staff, or visitor) will have different menu options. All options will be available to students, while visitors can access only the options that do not require database information. Individuals seeking access to database information (students, counselors, faculty, or staff) will be requested to identify themselves by entering their social security number and PIN. In a later phase of implementation, the kiosk will read bar-coded student I.D. cards to control information access.

General Information Module

This module provides institutional information such as campus maps, processes and procedures (e.g., how to register for classes), information about academic programs and facilities such as the bookstore and library. This information will be accessed in a hypertext mode. Some of these topics will later be enhanced to include multimedia modes of presentation. The objective is not to simply duplicate hard-copy information (such as the institution's catalog), but to take advantage of interactive access and multimedia facilities which are not possible with printed media.

Access to Student Records

This function provides the student with direct access to his/her database records, including demographic data, the student's transcript, current schedule, and fee statement. The information can be displayed on the kiosk and (optionally) printed.

Career/Major Choices

This module initiates a dialogue with the student regarding personal preferences (using factors similar to the Strong-Campbell Interest Inventory), academic strengths and weaknesses, and job-related factors that might be relevant to the student's career or major choice, such as job placement rates and average starting salaries. CWEST does a match of the student's responses with the characteristics of career programs and majors offered at the institution. The system returns a list of the best matches, ordered by confidence of the best fit, for the student's consideration. (The module considers only programs that are offered at this particular institution; it is not intended to be a comprehensive career-planning advisor.)

Referrals to Personal Counseling

The Referral module does not give advice on personal problems; its objective is to refer the student to a human counselor. However, in order to determine the most appropriate office or service where the student can get help, the module will engage the student in a dialogue to try to identify the actual nature of the problem. Referrals will include support services such as the developmental counseling staff, tutorial services, peer counselors, substance abuse counselors, the campus ministry, adult re-entry, financial aid counselors, and academic counselors.

Financial Matters

This module will assist the student with estimates of educational costs (tuition, books, parking, etc.), possible sources of financial assistance, and estimates of eligibility for aid from various sources. It will also educate students about the institution's processes related to financial aid and deadlines. The module will draw on information from the centralized financial aid system, and augment it with case-based knowledge.

Progress Check

The primary function of this module is to produce a degree audit report (list of remaining

courses and requirements for the student's degree). This module will also include a "what if?" function, which is sometimes called "degree shopping." The what-if function provides the student with a list of courses that would remain to be taken if the student were to change his/her major to some other program.

The degree audit reports and "what-if" facility are both components of a comprehensive information system that runs on the central computer; the CWEST system provides a convenient means for the students to access these reports.

Also, the project team will look at the feasibility of student performance trend analysis as an extension of the Progress Check module.

Credit Hour Load Recommendations

One of the primary causes of academic difficulty among non-traditional students is an unrealistic assumption about the number of credit hours they can handle along with non-academic commitments. Since these students are often in a hurry to complete their academic programs, it is common for those who are self-advised to overload themselves. For this reason, the Credit Hour Load module is one of the most significant in the system.

The reasoning of the Credit Hour Load module is based primarily on interactive responses from the student rather than database information. The module considers a number of factors including the student's employment, children at home, current grade point average, degree of support at home (encouragement or discouragement), the level of job-related stress, requirements imposed by financial aid, and whether the student has a compelling reason to graduate as soon as possible. The expert system also recognizes cases where it is imperative for the student to see a human counselor before registration. This is true for almost all students on probation, and cases where the student's financial aid package requires him/her to take more credit hours than CWEST determines that the student can handle.

The essence and logic of this module -- the factors and their relative weights -- were developed through intensive knowledge-engineering sessions with the project experts, and it reflects their extensive experience with real-world situations.

Course Recommendations

This module will provide recommendations for specific courses to take next term. It makes extensive use of course and student information from the central databases.

In developing a schedule recommendation, the module considers the course sequence recommended for the student's major, prerequisite courses, and the cognitive demands of individual courses in order to recommend a balanced schedule. The cognitive demands are obtained from course profiles, which are ratings of each course with respect to reading, writing, and computational difficulty, as well as the emphasis of the course on memorization, conceptualization, and problem solving, and the number of significant (time-consuming) projects or papers. After pondering all of these factors, CWEST returns an optimal schedule, along with at least three alternate courses.

IV. System Development

An early decision by the CWEST team was to employ an incremental design and development process with prototyping. The expert system shell that we selected for trial prototypes was ART-IM/MS-DOS from Inference Corporation. ART-IM provides a robust development environment, a rich set of language features, and convenient graphical user interface facilities. While the learning curve for ART-IM is challenging, we feel that it has been worth the investment in time and effort to become proficient in the use of the product.

The platform selected for deployment is IBM's Audio Visual Connection under OS/2 on IBM's recently announced kiosk, the Ultimedia Touch Activity Center.

V. Implementation Plans

The deployment of CWEST is planned as a three-phase process. Since we are exploring new territory in the application of expert systems technology, as well as in multimedia interface design for a kiosk, we feel that it is essential to thoroughly evaluate the system in terms of both the quality of CWEST's advice and the subjective and emotional responses of the students. For this reason, the first phase will include a subset of the modules and somewhat limited multimedia facilities, so that we can benefit from actual experience before implementing the entire system.

Scheduled for implementation late in the academic year 1992-93, the initial version will include:

- o credit hour load recommendations
- o financial matters
- o matching career/major options with interests
- o access to database information (read only)
- o general information about the institution

Kiosk facilities for the first version will include a touch-sensitive screen, graphics, sound, printer, and a link to the centralized student and course databases.

During academic year 1993-94, the CWEST team will evaluate the students' reactions to the first version and finalize the development of the second version. The second version will incorporate changes as needed and add additional functionality, which will include course recommendations, academic progress check, and referrals to personal counseling. The second version will also include additional technical enhancements -- specifically, the ability to read bar-coded student I.D. cards for improved security, and the use of motion video to provide general information about the institution and its academic programs.

During the third year (1994-95), the second release will be evaluated, and the system will be enhanced to provide actual on-line registration through the kiosk. The student can then engage in a dialogue with CWEST to determine an appropriate credit hour load and specific courses to take, and then register for classes and leave the kiosk with a schedule in hand.

VII. Conclusions

Using today's technology, the CWEST student counseling system will capture the expertise of the best and brightest counselors at the authors' institution, and make this knowledge readily available to students through kiosks located at convenient sites on and off campus.

While this application might seem to be specific to the educational environment, the "intelligent kiosk" approach has much broader implications. The approach could be relevant to almost any enterprise that produces complex products or services, where customers need assistance with the selection of options that will meet their personal needs and circumstances. By analogy, our complex products are academic programs and courses, and our confused customers are students.

With the CWEST system, we hope to demonstrate that the blend of expert systems technology, database access, and multimedia can result in a user-sensitive application that is functional, cost-effective, and engaging to use.

References

[1] Rash, Dennis. "Strategies for the Community College Counselor." *Community College Week.* September 16, 1991. p. 5.

[2] Ragusa, J.M. and Orwig, C.W. "Attacking the Information Access Problem with Expert Systems." *Expert Systems: Planning/ Implementation/Integration.* Winter 1991. pp. 26-32.

PACKAGED EXPERT SYSTEMS LANGUAGES: A COMPARISON

Ali Salehnia Bin Cong Sung Yun Shin Mehran Pournaghshband
South Dakota State University University of Arkansas
Brookings, SD 57007 Monticello, AK 71655

OPS5, CLIPS, LOOPS, LISP, and PROLOG are popular expert system languages. The person who works with these languages should be familiar with the syntax and inference engine of these languages. There are differences in their inference engine, syntax, price, portability, the kind of language that they are build on, and their complexity. For example OPS5 has a forward inference engine while PROLOG has a backward inference engine. Some of these languages are very similar such as OPS5 and CLIPS. Both these languages have forward inference engine, and their syntax is very similar. In this paper, the authors present not only the similarities and differences among these packaged expert systems languages but also the strengths and weaknesses of each.

SYSTEMATIZING A SYSTEMS PROJECT

S. K. SHAH
SETON HALL UNIVERSITY

ABSTRACT

Over the last few years, systems professionals have used several models and methodologies to deal with application backlogs and spiralling maintenance costs. Increasing attempts are being made to improve productivity and project quality. This paper describes a methodology for systems development and enhancement projects. It identifies major deliverables for management, user groups, and systems professionals to facilitate a clear understanding of their role and responsibilities. The process allows participants to select those activities needed to deliver the necessary functions within the required time frame and budget. It is expected that adherence to this methodology would alleviate tension and promote a better understanding between user groups and systems professionals.

INTRODUCTION TO GROUP DYNAMICS: A CRITICAL FACTOR IN SYSTEM ANALYSIS & DESIGN COURSES

Regina L. Smart
Southeast Missouri State University

ABSTRACT

Many colleges and universities today utilize team projects in system analysis and design courses based on actual system projects. This group approach is an attempt to introduce students to real-world technical experiences and competencies. In most instances, the experience also introduces students to real-world frustrations and the conflicts inherent in working relationships within groups. This paper discusses system analysis and design group project dynamics and related issues relative to group formation, size, cohesiveness, leadership, norms, productivity, evaluation, task characteristics, and individual roles. The author maintains that students will have a more positive group experience in system analysis and design courses if they are introduced to the inherent problems, frustrations, and benefits of group environments in addition to their traditional instruction in systems development.

ADAPTING A MS/MIS CURRICULUM TO A CHANGING ENVIRONMENT*

Thomas E. Sandman and Metta Ongkasuwan
MIS Dept., School of Business Administration
California State University at Sacramento
6000 J Street, Sacramento, CA 95819-6088
(916)278-6670, Fax (916)278-6757, E-Mail sandmant@csus.edu

Abstract. *There are several environmental forces which act to shape the curriculum at any given institution. This paper describes the curriculum reassessment process which is currently being conducted at a regional public university for its MS/MIS program. Both current and proposed curricula are presented. The paper presents discussions of these curricula from the perspective of an environmental forces framework.*

1. Introduction.

Curriculum reassessment is a necessary and vital activity of any educational institution. This activity is necessitated by changes in the underlying domain knowledges, changes in the goals of the program, changes in the constituency of the program, and/or changes in the social/political/economic environment of the institution. The purpose of this paper is to describe how several of these forces have triggered a reassessment of the Master of Science in Management Information Systems (MS/MIS) curriculum in our institution. This reassessment has led to a revised curriculum which is in the process of being implemented. The primary goal of this process is to redefine the program to ensure the highest level of educational quality, while conforming to the growing set of strict resource constraints that are facing every public higher education system in the United States.

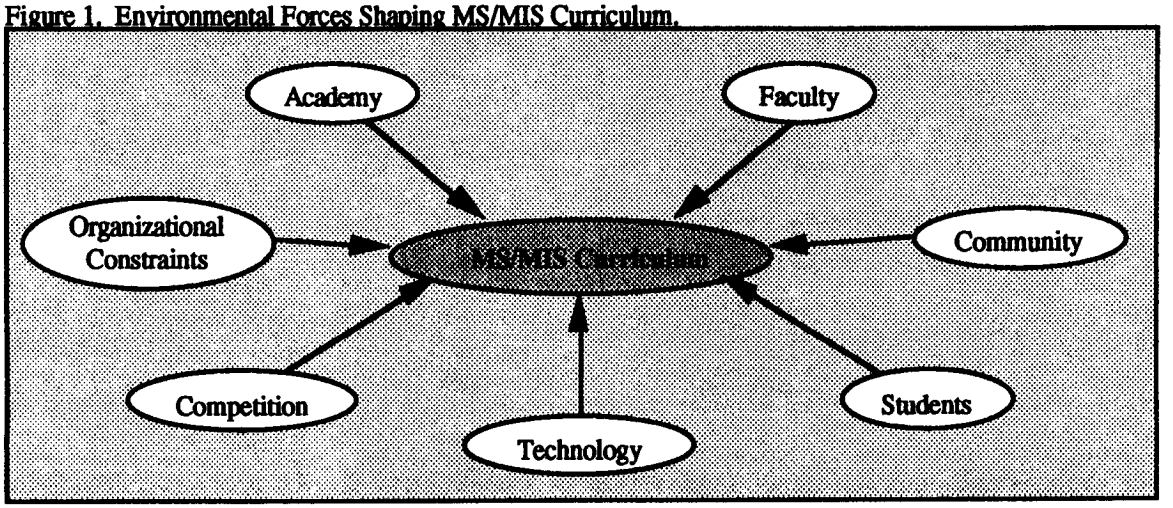
There are approximately 160 master degree level graduate information systems programs in the United States and Canada [1]. While the focus of this paper is on one particular MS/MIS program, it is believed that there are many similarities between the issues faced here and those facing the other 159 graduate programs in MIS. Our presentation begins with a discussion of the major forces which act to shape our curriculum. In §3, this framework is used to describe the current status of the MS/MIS program. Both opportunities and weaknesses are identified through this process. Based on this information, a revised curriculum is proposed in §4. Implementation issues associated with the proposed program are presented in §5. This paper concludes with a summary of our attempt to define an efficient, quality MS/MIS program which is responsive to the environmental forces presented.

2. Environmental Forces.

The environmental changes which trigger curriculum reassessment correspond to seven principal forces which shape any MS/MIS curriculum (see Figure 1). These forces are: the Academy, the Faculty, the Community, the Students, the Technology, the Competition, and the Organizational Constraints of the institution. Each of these forces is discussed below.

The strongest of these forces is that of the Academy. This represents the requisite common body of knowledge which is prescribed for any MS/MIS graduate. There are several models which are defined to articulate this body of knowledge. These models stem from efforts by professional organizations to enhance and support their members. Both the Association for Computing Machinery (ACM) and the Data Processing Management Association (DPMA) have the curriculum guidelines. Curriculum development has also been directly supported by major corporations (e.g., IBM's multi-million dollar Management of Information Systems Grant Program of the late 1980s [2]). While these

Figure 1. Environmental Forces Shaping MS/MIS Curriculum.



**This paper represents the views of the authors and may, or may not, be endorsed by the rest of the Faculty of the MIS Department.*

organizations may not have formal accrediting procedures, they identify the major topics which form the common body of knowledge for masters level information systems programs.

The next force is the Faculty of the institution. Any curriculum is only as strong as the faculty who conduct it. One impetus behind the IBM Grant Program was the dearth of qualified IS doctorates (individuals with a D.B.A. or Ph.D. in the information systems field). Currently, many MIS faculty do not have their formal training in information systems, but have moved from related fields. The curriculum may be restricted by the availability and number of qualified professors.

The Community is the third force which shapes the curriculum. Our perspective of the community is focused on industry. MS/MIS graduates either continue their education in doctoral programs or enter the work force. If our MS/MIS curriculum is not responsive to the needs of our consumers (e.g., local industry), they will not hire our graduates. Our curriculum must prepare students for either path, which are, at times, quite divergent.

The Students also impact the MS/MIS program. Student demographics, student preparation, and the overall student population greatly affect the number and frequency of course offerings. Students without prior MIS backgrounds must successfully complete the program prerequisites, which may substantially increase the costs borne by the student (in both time and money).

Technology is the fifth force which shapes the MS/MIS curriculum. Information technology is extremely dynamic. Major new technologies are being developed and are becoming paradigmatic of organizations (e.g., Expert Systems [3], Electronic Meeting Systems [4]). At some point, it may be necessary to alter the content of a course or program to address the issues associated with the paradigm. This is a technology driven field. Therefore, there is a necessary orientation towards emergent technologies.

Competition for students from other universities and training programs is also a factor. Students may elect to enroll in non-AACSB accredited programs if they believe that their goals will be met at a lower cost (temporal or monetary). This type of competition necessitates efforts to maintain the visibility of the program, and increase the awareness of potential students concerning the program.

The final force shaping the MS/MIS curriculum is Organizational Constraints. This force is, perhaps, the one that most limits our potential curriculum. Organizational constraints consist of all of the factors affecting resource allocation. Higher education systems across the country are facing severe budgetary restrictions. This increases the intraorganizational attempts to exert social power and influence, which is determined, in a large part, by control over critical resources [5]. As these budgetary restrictions continue, attempts will be made to obtain more control over important resources [6], such as course full-time equivalent enrollment (FTE).

These seven forces from Figure 1 will impact each MS/MIS program differently. MS/MIS program can themselves be viewed as systems where: the primary input is the student, the curriculum represents the primary processing, and the graduates are the output. These forces form the environmental boundary, and each of them impacts the quality of the output, the quality of the input, and the quality of the processing. The curriculum reassessment process is the feedback loop which will provide the ability for the system to adapt to its environment.

3. Current Status.

The current MS/MIS curriculum is modelled after the ACM and DPMA guidelines for information systems majors. This program is designed to accommodate the students who have an undergraduate background in management information systems. However, students not possessing MIS training may participate in the MS/MIS program after completing the following program prerequisites:

Prerequisite Courses:	Hrs:	ACM	DPMA*
Intro to Applic Prog.	3		CIS 1
Advanced COBOL.	3		CIS 3
Analysis and Design	3		CIS 4
Database	3		CIS 5
Management Science	3		
Prereq. Hours	15		

* DPMA codes shown throughout are actually the undergraduate program codes.

The MS/MIS program requires 30 units beyond the prerequisite courses. Each of the following courses is required:

Core Courses:	Hrs:	ACM	DPMA
Project Mgt. & Implementation	3	IS 10	CIS 8
Data Communications.	3	IS 6	CE 4
Adv Analysis & Design	3	IS 5	CIS 7
Database Administration	3	IS 4	CE 15
Major Applications of MIS	3	IS 7A	
Managing the IS Resource.	3	IS 9	CIS 9
Policy Formulation.	3		
Internship in MIS	3		
Master's Thesis or Project.	3		
Core Hours	21-27		

Students may choose from the following optional courses in order to complete the 30 unit requirement for the MS/MIS:

Elective Courses:	Hrs:	ACM	DPMA
Business Programming	3	IS 1	CIS 2
- or- Advanced COBOL.	3	IS 1	CIS 2
Topics in MIS	3	X 1	
Evaluation Hardware/Software	3		CE 14
Intermediate Statisticss.	3		
Elective Hours	3-9		

While this curriculum may appear to be adequate, it is not. All seven forces are pushing for a change in the curriculum. From the Academy perspective, both the ACM and the DPMA are in the process of revising their respective curricula [1]. This will require evaluation of program course content in order to expand our coverage to the new topics, as well as those not previously covered.

Although 64% of the Faculty teach in the MIS area, approximately 21% of the Faculty within the department have a Ph.D. in MIS. A sizeable portion (about 42%) of the faculty has been with the institution almost 20 years. As these faculty retire, they will be replaced by new IS doctorates. It is projected that the demand/supply ratio for IS doctorates will drop from 4.1 to 3.6 in the 1992-93 academic year [7]. This drop in positions will actually help us draw more highly qualified candidates for recruitment. With fewer doctoral granting institutions hiring, regional masters level institutions will benefit. Since our institution is prohibited from offering doctoral degrees, a strong vital MS/MIS program will be an asset in faculty recruitment activities.

While there has been no degradation in the reception our graduates get from the Community, there are aspects of the current program which need to be revised in order to maintain the attractiveness of our graduates. These include less reliance on COBOL platforms, and more emphasis on emergent technologies (e.g., CASE, KBDSS). This region has a very (relatively) healthy business climate. There is a growing population of IS related firms within the area which should increase the demand for our graduates.

Students are a major factor. Most of our students work full time and take evening classes. However, despite good employment prospects in a suppressed economy, our MS/MIS enrollments have been low recently (see Figure 2). This may be a result of a general lack of awareness of the program. Another limitation is the number of prerequisite courses needed for those without prior MIS training. The fact that this may substantially increase the costs borne by the student (in both time and money), is often mentioned by our students. A proposed increase of 40% in student fees has the potential of negatively affecting our enrollments.

The field of Management Information Systems is extremely dynamic. The MIS Department is continually examining its MS/MIS program in order to provide the appropriate blend of theories and applications, for a wide variety of MIS topics. There is an ongoing review of the program with the goal of assuring a focus on emergent technologies. However, this must be balanced by a strong foundation in MIS theory.

Our Competition is any alternative to an AACSB accredited MS/MIS degree in the region. This includes choices to forgo advanced education, but our concern lies primarily with those students who choose to attend small private sector less accredited universities. Again, any loss of students to the competition is seen mainly as an awareness problem. Although our program may take longer to complete, the MS/MIS degree from our institution would have much broader recognition than that of any alternate program in the region. Furthermore, the principle institutional competition does not have AACSB accreditation.

By far Organizational Constraints of the institution have the greatest impact on the current curriculum. These constraints follow three lines. First, are the constraints which are resource allocation related. The university has had two consecutive years with budget reductions, with the 1991-92 budget representing 8 - 10% less purchasing power than the previous year's [8]. Without higher FTE, there may be difficulty offering even the core courses for the MS/MIS program. This is exacerbated by our suppressed program enrollments. The second set of constraints which inhibits our ability to readily change our curriculum is the time frame for implementing changes. The university publishes its course catalog every other year. This means that any changes that are approved in the next year will not take effect until the 1994-1995 school year. Another constraint is that the MS requirement of 30 program hours can not be reduced. The final set of constraints is the social power struggle for FTE with the Computer Science Department. As fiscal resources have diminished, nearly every new course proposal or existing course modification tends to be challenged more rigorously by Computer Science. This makes the implementation of the more technical ACM and DPMA courses rather difficult.

4. Proposed Curriculum.

Figure 3 presents our perspective of the common body of MIS knowledge that our graduates must have (after Vitarali's model of a systems analyst's knowledge base [9]). The programming languages and computer system organization domains require each student to master programming skills and knowledge (e.g., programming methods, standards, modes, and languages) supported by a variety of computer system environments (e.g., Unix, Vax, IBM). The systems analysis, systems design, database design & management, data communication & networks, and intelligent systems &

Figure 2. MS/MIS Enrollments, by Semester.

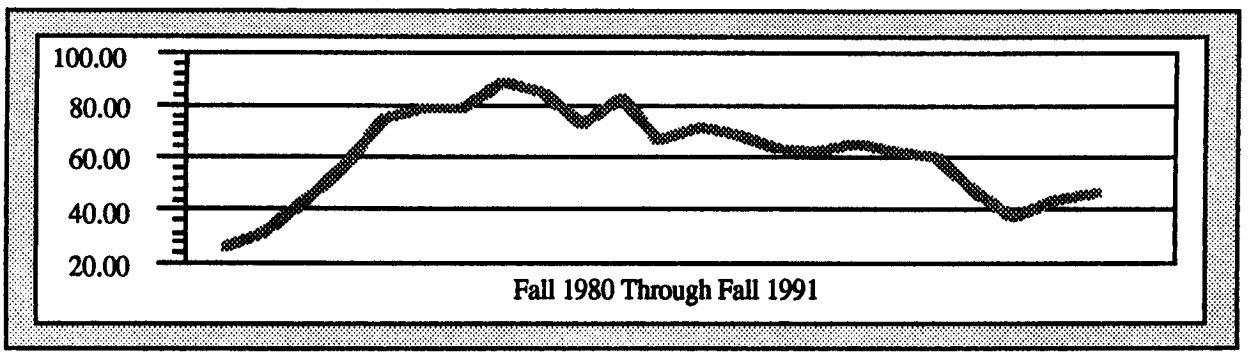
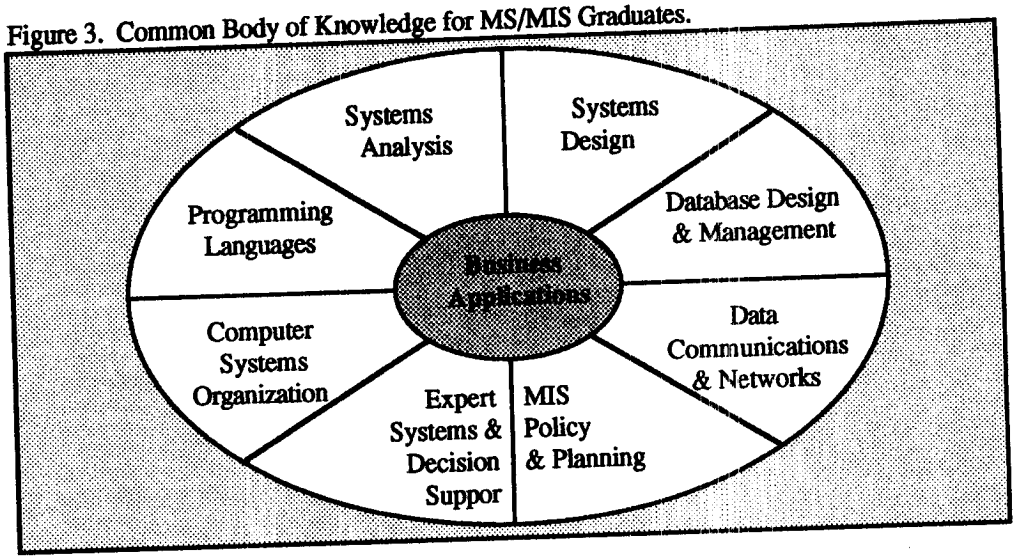


Figure 3. Common Body of Knowledge for MS/MIS Graduates.



decision support domains require each student to acquire and maintain specific knowledge related to concepts, methodologies, techniques, tools, approaches, and underlying management principles. The MIS policy & planning domain requires students to acquire the basic IS strategic planning knowledge and extended skills to formulate IS strategic plans and implement them under a constantly changing environment. The core domain knowledge concerns business applications. This domain requires each individual to acquire knowledge and experience in business functional areas (e.g., accounting, finance, production) and organizational perspectives (e.g., organizational behavior, strategy, structure).

The proposed MS/MIS curriculum represents an attempt to provide this common body of knowledge while balancing the external curriculum forces. This is necessary in order to attract students and provide them with the best MS/MIS program possible. The first revision concerns prerequisites. The prerequisites have been drastically reduced. Two new, highly intensive graduate level courses that will provide students with fundamental MIS knowledge are being introduced. The first course covers analysis and design methodologies, along with the support tools (e.g., CASE). The second course focuses on structured programming, data structures, and database management. When taken as a set, these two courses will provide a richer background than the five undergraduate courses which have been serving as the prerequisites. Therefore, students not possessing MIS training would have to complete the following program prerequisites:

Prerequisite Courses:	Hrs:	ACM	DPMA
Information Systems I.	3	IS 3	CIS 2/4
Information Systems II.	3	IS 1/2	CIS 1/3
Prereq. Hours	6		

The MS/MIS program still requires 30 units beyond the prerequisite courses. The proposed curriculum changes the core of the MS/MIS program to a narrow set of courses, which will allow students greater flexibility in choosing electives to complete their program. For the requirement of a 'culminating experience,' the option of a comprehensive examination has been added. The following courses represent the proposed core:

Core Courses:	Hrs:	ACM	DPMA
Data Communications.	3	IS 6	CE 4
Adv Analysis & Design	3	IS 5/8	CE 11
Database Design & Managment	3	IS 4	CE 15
MIS Policy.	3	IS 9	CIS 9
Master's Thesis or Project	3	X 2	
or Comprehensive Exam.	1		
Core Hours	13 - 15		

Students will then choose from the following optional courses in order to complete the 30 unit requirement for the MS/MIS:

Elective Courses:	Hrs:	ACM	DPMA
Business Programming.	3	IS 1	CIS 3
- or- Advanced COBOL.	3	IS 1	CIS 3
Expert Systems and DSS	3	IS 7AB	CE 9
Topics in MIS	3	X 1	
Computer System Organization	3	IS 8	CE 14
Decision Modelling	3	IS 7	
Internship in MIS.	3	IS 10	CIS 8
Elective Hours	15 - 18		

5. Implementation Issues.

This new curriculum provides a better match to the environment as defined by the seven forces. From the Academy perspective, the new curriculum provides a more consistent coverage of the ACM and DPMA curriculum guidelines.

Our Faculty is prepared for the changes, and eager to see them implemented. The greatest complexity introduced by the new curriculum is the comprehensive examination. Issues of examination content, and grading need to be addressed, although no resistance to the comprehensive examination option is anticipated. The revised curriculum will help to attract new faculty. The concerted efforts by the Faculty in this reassessment process also make the department attractive.

The modernized curriculum should also strengthen the reception our graduates get from the Community. Efforts to make the Community aware of our program need to be continued. Liaisons need to be forged with local industry and foster the symbiotic relationships that should naturally exist. The Department is looking to further expand its presence in the work place of the major employers through broader use of televised course offerings.

Students will receive the greatest benefit from our curriculum revision. Many of our students will directly benefit from the streamlined prerequisites. The lack of program visibility is being addressed with new brochures, although the brochures must reflect the current curriculum. These brochures will be updated to reflect the changes, if and when they are approved. With fewer prerequisites and the comprehensive examination, the MS/MIS degree from our institution can be obtained within 2 years. Furthermore, by offering the comprehensive examination, students will no longer have to be concerned about having a poor thesis or project prevent their completion of the program.

Our ability to react to the Technology force is largely a function of resources. The MIS Department is continually examining its MS/MIS program in order to provide a cost effective introduction of new technologies. This reflects the goal of assuring a focus on emergent technologies. However, with limited resources, this remains an area of great concern.

Since our Competition is principally limited to a small private sector university, the new curriculum will strengthen our position. Again, any loss of students to the competition is seen mainly as being caused by the student not knowing about our MS/MIS program.

The greatest challenge to our proposed curriculum stems from Organizational Constraints of the institution. The social power struggle with the Computer Science Department is not trivial. The proposed curriculum contains three new course proposals (including the comprehensive examination) and five course modifications that require approval at the Department, School, and University levels. No problems are expected until the proposals reach the University level, where it is suspected that they will be challenged vigorously by Computer Science.

6. Summary.

The purpose of this paper has been to describe how environmental forces trigger and shape curriculum reassessment. The focus has been on seven principal forces which shape any MS/MIS curriculum. These forces are: the Academy, the Faculty, the Community, the Students, the Technology, the Competition, and the Organizational Constraints of the institution. There may be several that have been overlooked, and perhaps there are some which exist yet do not apply to the MS/MIS program here. The primary goal of this process is to establish programs which ensure the highest level of educational quality, while conforming to the growing set of strict resource constraints that are facing every public higher education system in the United States.

Responsibility for ensuring curriculum excellence extends beyond the faculty to the students, the campus, the university, and the regional community. Curriculum reassessment must be done periodically in order to meet the needs of our constituents in applying and advancing information systems technologies in the local communities. The continual improvement of instructional methods, high technology devices, equipment, and materials will provide opportunities for both students and faculty to develop and maintain new skills and knowledge. Institutional support should be provided by establishing educational goals which support the rapidly changing business and technology environment. The effort of faculty towards curriculum review should be acknowledged as a recognized criterion for their performance evaluations. The institution and the community should ensure a continual reinvestment in information technology (equipment and materials) which would support teaching and research in information systems.

7. References.

[1] DeGross, JI, Davis, GB, and Littlefield, RS, 1992 Directory of Management Information Systems Faculty in the United States and Canada, MISRC/McGraw-Hill, 1992.

[2] --, IBM MoIS Syllabus Book, Information Systems Research Program, John E. Anderson Graduate School of Management at UCLA, Los Angeles, CA, 1990.

[3] Blanning, RW, "Expert Systems as an Organizational Paradigm," Proceedings of the 8th International Conference on Information Systems, Pittsburgh, 1987, pp. 232-241.

[4] Eck, R, Goul, M, Philippakis, A, and Richards, S, "Group Operating Systems for Decision Factories of the Future: An Extended Relational GDSS Architecture," Proceedings of the 22nd Hawaii International Conference on Systems Sciences - Volume III, 1989, pp. 280-290.

[5] Pfeffer, J, Organizational Design, AHM Publishing Corporation, Arlington Heights, IL, 1978.

[6] Pfeffer, J, and Salancik, GR, The External Control of Organizations: a Resource Dependence Perspective, Harper & Row, New York, NY, 1978.

[7] Jarvenpaa, SL, Ives, B, and Davis, GB, "Supply/Demand of IS Doctorates in the 1990s," Communications of the ACM, Volume 34, Number 1, 1991, pp. 86-99.

[8] Harrison, M, "The 1991/2 Budget at _____ University," Internal Memo, February, 1992.

[9] Vitarali, NP, "Knowledge as a Basis for Expertise in System's Analysis," MIS Quarterly, September, 1985, pp. 221 - 241.

ATTITUDES AND COMPUTING ENVIRONMENTS OF EMPLOYERS OF ENTRY-LEVEL COMPUTING POSITIONS

William Myers
Belmont Abbey College
Belmont, NC 28012-2795

ABSTRACT

A survey of the views of employers of CIS graduates concerning the competencies they sought in graduates and concerning the current and future computer environments was conducted in the Charlotte metropolitan area. The results are compared with similar studies. The findings revealed that:

1) **interpersonal, communication, and system analysis skills were highly valued by the employers.**
2) **the majority program in COBOL, RPG, or various database and plan to do so in five years.**
3) **PCs are an important component of the computing environment.**

INTRODUCTION

CIS graduates are entering constantly changing computing environemnts. As a consequence, CIS programs need to be continually reevaluated to prepare their graduates for the environments in which they will be expected to function.

During 1991, the Computer Studies faculty at Belmont Abbey College conducted a review of its programs that include a CIS major and a CS and MIS minor. This review consisted of three parts: an outside evaluation, a survey of recent CIS graduates, and a survey of likely employers in the Charlotte metropolitan area. During this year and next, these results will be used to produce recommendations for modifications and changes in the department's programs.

This paper presents results of the survey of employers. Its purposes were two fold: 1) to determine what skills and knowledge areas likely employers were to consider important for entry-level positions, and 2) to determine these companies' computing environments. The survey was based on a survey of employers in the St. Louis area conducted by Dr. Mary Sumner of Southern Illinois University [8]. Her survey was supplemented with questions about the use of programming languages and operating systems and about additional topics contained in Belmont Abbey's curriculum.

The survey was sent to 303 employers of computing professionals in the Charlotte metropolitan area that had contact with the Cooperative Education Program at Belmont Abbey College. (Cooperative Education arranges for internship-like work experiences for students of the College in a variety of area businesses.) Fifty-four (54) of the surveys were returned (for a response rate of 18%) of which fifty-one (51) were usable.

RESULTS

The forecasted recession and the decline in job offers to our graduates in 1990 and 1991 prompted us to ask employers about their demand for entry-level computer personnel and their educational requirements. A profile of the size of the companies and their computer departments are in Tables 1 and 2.

TABLE 1 — SIZE OF FIRMS

Size of Firm	Percentage of Firms
5000 or more employees	0%
1000 to 4999 employees	14%
250 to 1000 employees	29%
Under 250 employees	41%
No response to question	16%

TABLE 2 — SIZE OF COMPUTER DEPARTMENT

Department Size	Percentage of Firms
100 or more employees	6%
50 to 99 employees	6%
Under 50 employees	74%
No response	14%

In spite of employment trend statistics

from the Department of Labor trends that indicate that programmers and system analysts are among the fastest growing professions [3], our experience for 1990 and 1991 was that our graduates were taking longer to locate positions with local firms. In these two years, most did not find employment until the end of the summer. In addition in 1991, there were concerns about the length and depth of the then predicted short recession. The survey results seem to indicate that in the Charlotte area the number of new positions in the computing professions are declining. Table 3 displays the responses to questions concerning past, present and future needs for entry-level personnel. Table 4 displays the number of firms desiring or requiring a CIS degree for entry-level personnel.

TABLE 3 -- ENTRY-LEVEL PERSONNEL NEEDS

	1990	1991	1992 (proj.)
Technical Support Personnel			
Number firms needing	7	2	6
Average number needed for these firms	3.4	1.5	1.5
Total persons needed	24	3	9
Technical Programming Personnel			
Number firms needing	13	8	16
Average number needed for these firms	3.3	2.8	1.8
Total persons needed	43	22	29
Technical Systems Analyst			
Number firms needing	3	2	3
Average number needed for these firms	3.7	1.5	1.3
Total persons needed	11	3	4
Technical Operations Personnel			
Number firms needing	11	6	10
Average number needed for these firms	1.5	1.8	1.2
Total number needed	17	11	12

TABLE 4 -- DEGREE REQUIREMENT FOR ENTRY-LEVEL PERSONNEL

Position	Number Who Desire CIS Degree	Require
Technical Support	8	7
Programming	8	13
Systems Analysis	1	8
Operations	8	3

The computer professions are rapidly evolving and expanding. The employers were asked to identify positions of emerging importance to their firms. Table 5 lists in order the positions listed by at least 10% of the respondents.

TABLE 5 -- POSITIONS OF EMERGING IMPORTANCE

Category	Number of times listed
Networks	13
Programming	8
System Development	8
Database	6
Microcomputer Support	5
Data Communications	5

TABLE 6 -- COMPETENCIES NEEDED FOR SUCCESSFUL JOB PERFORMANCE

Competency	Average I	Average II
Interpersonal Skills	4.6	4.3
Written Communication	4.2	3.9
Speaking Skills	4.1	3.8
Basic Systems Analysis	4.0	3.7
End-User Support	3.9	3.6
General Business	3.8	3.6
COBOL Programming	3.7	2.5
Microcomputer Software	3.7	3.2
Basic Database Design	3.7	3.2
Data Communications	3.7	3.1
Advanced COBOL	3.6	2.4
Operating Systems	3.5	3.3
LAN Design/Management	3.4	3.0
Hardware Concepts	3.4	3.1
JCL	3.2	2.5
PC Software Packages	3.2	2.9
Advanced Database	3.2	2.7
Advanced Systems Analy.	2.9	2.3
Basic Accounting	2.9	2.7
Programming in 4GL's	2.9	2.3
CASE Tools	2.8	2.2
Decision Support Tools	2.7	2.3
Quantitative Methods	2.7	2.5
Expert Systems	1.8	1.5

Employers rated the competencies they sought in recent CIS graduates. The average of their ratings are reported in Table 6 on a five point scale (5 very important to 1 not important). The first column excludes those who checked not

applicable or did not answer the question; the second column counts either of those responses as a 1, not important. The most important competencies identified were good interpersonal and communication skills and basic systems analysis.

Responses as to the languages presently used and those expected to be used in five years are listed in Table 7. All responses greater than 10% are included. The employers identified COBOL and RPG as their most used languages both now and in the near future with C expecting to be of increasing importance. A large number use database languages and expect to continue to do so, with SQL becoming more important. Various fourth generation languages are not widely used, nor are they expected to be (although many of the database packages have many of the features of fourth generation languages).

TABLE 7 -- LANGUAGE USAGE

Language	% Presently Used	% Expected in 5 years
COBOL	45%	29%
RPG	25%	16%
BASIC	10%	4%
C	8%	16%
SQL	2%	10%
Other Database Languages	33%	20%
Fourth Generation Languages	10%	10%

TABLE 8 -- OPERATING SYSTEMS USAGE

Operating System	% Presently Used	% Expected in 5 years
MS/PC-DOS	37%	12%
IBM Mainframe (MVS, VM, etc.)	22%	16%
IBM Minicomputers (Sys 36, 38, AS-400)	20%	24%*
VMS (DEC)	18%	16%
UNIX (various versions)	10%	25%
Novell NetWare	10%	12%
OS/2	6%	20%

* -- These are all OS/400.

Responses to the operating systems presently used and those expected to be used in five years are listed in Table 8. All responses greater than 10% are included. These firms are primarily using IBM PCs and compatibles and IBM mainframe and minicomputers and intend to continue to do so. UNIX and OS/2 usage is projected to increase, supplanting most of the present use of MS/PC-DOS.

For the complete results of the survey and a copy of the survey instrument, see [5].

COMPARISONS WITH PREVIOUS SURVEYS

Seven previous surveys of employers of entry-level computer positions are used for comparisons, one each conducted in Georgia [9], Kentucky [1], the Pittsburgh area [6], the Spokane, Washington, area [7], and the St. Louis area [8], as well as a previous study in North Carolina [4], and one nation-wide study [2]. The present survey consists primarily of responses from small firms (under 1000 employees) whereas four of the other seven consisted primarily of responses from large firms (1000 or more). Most of the few differences between this and the other surveys can be explained by the differences in the size of the firms surveyed.

The four previous surveys of large firms ([1], [6], [8], and [9]) as well as one of the others ([3]) found more emphasis on programming (especially in COBOL) than this survey. In this as well as those five surveys, programming in COBOL was one of the most important competencies. COBOL programming was important to many of the firms in this survey and ranked high among those firms, but nearly half of the firms surveyed did not feel that this skill would be considered when hiring entry-level employees. [7] found an even larger percentage of respondents for which COBOL programming ability was not an important skill for their entry-level personnel.

Other differences between the present survey and the seven surveys include a lower ranking in this survey (with most rating the skill as not applicable) of

knowledge of JCL and fourth generation languages and a much higher ranking here for knowledge of micro-computer software, data communications (with the exception of [1]), and local area networks. [7] found similar low rankings for fourth generation languages and high rankings for microcomputer software and packages; it did not ask about the other areas.

[4], [1], [6], and [2] also inquired about programming languages used. In both this and all but [1] (where PL/I was the leader), COBOL was the leader by a wide margin. RPG was the second most used language in this survey and in [4]; it was highly ranked in [1] and in [2] but listed on [6]. [7] found that RPG and BASIC were the highest ranked languages, but all languages were considered to be unimportant by a large majority.

All of these differences can be explained by the differences in the firms' sizes. Smaller firms are more likely to use personal computers and minicomputers. In the PC world, COBOL and JCL are not as important; packages, data communications, and LANs are of more importance. RPG is an important tool with IBM minicomputers, but not as important in the mainframe world. Larger shops also do more in-house programming; thus they rank higher programming skills as in the four surveys of larger sized firms. [7] had an even higher percentage of respondents using PC environments than this survey; however, even the size difference does not explain the great majority felling that no language skills were important when hiring entry-level personnel. This result differs with all of the other surveys.

On several items there are agreement. The most important items (those with a ranking of 4.0 or more) in this survey, namely interpersonal skills, writing skills, speaking skills, and basic systems analysis, were in the most important five items in [8], were all highly ranked in [1], were mentioned the most in personal interviews in [7], and were in the top ten (out of twenty skills surveyed) in each of the three entry-level job categories in [9]. The first three of these items were in the most important six items in [6]. All these surveys agree that these four skills are among the most important skills for entry-level computer personnel. ([4] did not inquire about system analysis skills, although communication and interpersonal skills were the essence of its top two desired skills.)

There was also agreement between this survey and [8] on the emerging positions. All except programming were included in the top items in the previous study, and management of MIS (which was second in [8]) did not appear in the top responses in this survey. The other items were in the same relative positions. Once again these differences are due primarily to the differences in the companies' sizes. Smaller companies have less of a need for management, and they also appears to be seeing the need to design and program more of their own applications, as shown by the high number of responses for programming and system development.

CONCLUSIONS

The recession has had an impact on the number of entry-level positions in the Charlotte area. The number of firms hiring such personnel and the average number hired dropped drastically from 1990 to 1991. The projections for 1992 are closer to the 1991 levels than to 1990.

The majority of positions for entry-level programmers and nearly all the positions for entry-level analysts require a degree. As well, the majority of firms hiring entry-level technical support and operations personnel find it desirable that those persons have degrees. These firms have found that college educated computer students are of value in the firm's computer operations.

In the Charlotte area, the emerging computing positions are in networking, programming, and systems analysis. Firms are downsizing to networks and connecting personal computers. The firms also see an increasing need for custom programming and

for systems designed to meet their exact needs. COBOL and RPG are and will continue to be the major languages used; database languages are also important, with SQL increasingly important. The use of C is predicted to increase to a level matching RPG. Microcomputer operating systems are used by more than half the firms surveyed, a percentage that is expected to rise. Only microcomputer and AS/400 systems are projected to increase in use, with UNIX and OS/2 projected to be preferred microcomputer operating systems.

CURRICULUM IMPLICATIONS

Several curricular changes at Belmont Abbey have been supported by the survey results. A decision was made to retain an introductory unit on systems analysis in the introductory sequence. As well, the systems analysis sequence has been made more rigorous with the introduction of a broader range of tools used in industry, including CASE. In addition, a general introductory course for all majors was redesigned into an introductory course for nonmajors that concentrates on micro-computer packages and an introductory course for majors that contains more about computer hardware and operating systems, including introductions to MS/PC-DOS and UNIX. As well, an adjunct instructor was located to teach a course in networking and data communications. We have had such a course in the catalog, but no one in the department had the expertise to offer such a course. The course was offered for the first time this summer. The survey has also prompted the department to reinstate an RPG class and to encourage our majors to minor in business. (See [10] for a discussion of the continuing need to teach RPG due to the popularity of the AS/400.) Other changes may also be made after studying the survey of graduates based on the combined results of the surveys.

REFERENCES

1. Albin, Marvin, and Robert W. Otto. The CIS Curriculum: What Employers Want From CIS and General Business Majors. The Journal of Computer Information Systems, 27:4 (Summer, 1987), 15-19.
2. Bauman, Ben M., J.K. Pierson, and Karen A. Forcht. Business Programming Language Preferences in the 1990s. The Journal of Computer Information Systems, 32:1 (Fall, 1991), 13-16.
3. Franchi, Mandeline. IS Operations: The Ecomomy's Impact on Your Department. Datamation, 35:7 (April 1, 1989), 97.
4. Hunter, James W. What Topics Employers Think Should Be Included in CIS Courses. The Journal of Computer Information Systems, 27:3 (Spring, 1987), 23-26.
5. Myers, William H. The Job Market for Entry-Level Computer Professionals in a Southeastern City. The Journal of Computing in Small Colleges, 8(1993), to appear.
6. Pollack, Thomas A. The MIS Curriculum: Which Competencies Are Really Important to Business Professionals. Proceedings, ISECON'90, Chicago, 1990, 117-122.
7. Reitsch, Arthur G., and Frank E. Nelson. The Use of Industry and Student Perceptions in the Redesign of an MIS Curriculum. Interface, 12:3 (Fall, 1990), 6-13.
8. Sumner, Mary, Robert Klepper, and Robert Schultheis. An Assessment of the Attitudes of Graduates and Employers Toward Competencies Needed for Entry- Level MIS Positions. Proceedings, ISECON'90, Chicago, 1990, 129-134.
9. Walton, Hugh J., Dale Young, Shaila Miranda, Barry Robichaux, and Ron Seerley. Requisite Skills for New MIS Hires. Data Base, 21:1 (Spring, 1990), 20-29.
10. Winter, Charles R. Where Are the AS/400 Programmers? Information Executive, 3:3 (Summer, 1990), 19-22.

I wish to thank Dr. Mary Sumner for allowing me to use her survey instrument as the basis of my instrument, Mr. Carl van Orden for providing the list of local employers of CIS personnel, and Belmont Abbey College for financial assistance in conducting the survey.

A MARKET RESEARCH STUDY FOR GRADUATE EDUCATION IN MANAGEMENT INFORMATION SYSTEMS

Mary Sumner
Southern Illinois University at Edwardsville

BACKGROUND

The MIS curriculum is in a constant state of change. The changing needs of students, the changing expectations of industry, and the changing nature of the MIS profession itself are factors which dictate a continuing review of the focus of MIS curricula. One of the challenges of curriculum development is defining the market for a particular program and developing competencies which support the objectives of that market.

At the graduate level, two different markets exist for graduate education in Management Information Systems within the Business curriculum at Southern Illinois University at Edwardsville. The first of these markets consists of information systems professionals seeking advanced degree opportunities. The second market consists of functional area managers who are assuming responsibilities for information systems planning and development and who seek a graduate program with a strong emphasis in information technology planning and management. The Master of Science in Management Information Systems serves both of these markets. In Spring 1990, the Management Information Systems faculty decided to conduct a market research study to determine the needs of both markets and to identify curriculum competencies in a more focused manner.

OBJECTIVES OF THE STUDY

The major objective of the market research study for graduate education in information systems was to answer several questions:

1. What are the motivations for graduate study in information systems?

2. Are MIS professionals and functional area professionals interested in a graduate program with a business emphasis, a technical emphasis, or both?

3. What are the managerial competencies, technical competencies, and management of information systems skills desired by MIS professionals and by functional area managers in a program of graduate study?

4. What factors influence the choice of a graduate program?

METHODS FOR CONDUCTING THE STUDY

The population for this study included graduates of Business programs at Southern Illinois University at Edwardsville with three to five years of experience. A sample of 50 respondents was selected from this population. Of these respondents, 25 held positions in the information systems field, and the other 25 worked in functional areas, such as Accounting, Marketing, Finance, and Personnel Administration.

The 25 respondents who were information systems professionals reported a variety of position titles:

Table 1: Position Titles of MIS Respondents

Position Title:	Percentage:
Programmer/Analyst	35%
Manager, Information Systems Development	24%
Senior Systems Analyst	15%
Systems Administration (incl. Security)	8%
Manager, PC Systems	4%
Other	14%

The functional area managers had a wide variety of job titles:

Table 2: Job Titles of Functional Area Managers

Position Title:	Percentage:
Area Manager	20.3%
Sales Manager	8.4%
Senior Cost Analyst	8.4%
Administrative Asst.	8.4%
Admin. Manager	8.4%
Finance Manager/Analyst	8.4%
Contracts Admin.	4.2%
Account Manager	4.2%
Other	29.3%

Each of the respondents participated in a telephone interview which covered the market research questions in approximately eight to ten minutes. The telephone interviews were conducted by a market research analyst and a graduate assistant.

FINDINGS OF THE STUDY

A number of interesting issues emerged from the results. These issues included the importance of an advanced degree, the motivations for pursuing a Master's program, the most preferable type of degree (MBA vs. M.S degree), the importance of technical vs. managerial skills, and the major technical and managerial competencies to be developed in an MIS program.

Importance of the Master's Degree

Many of the respondents felt that a Master's degree was essential for career advancement. This was true of both the MIS professionals and functional area professionals. As you can see in Table 3, about 65 percent of the MIS professionals viewed a Master's as essential for career advancement. Work experience is also essential. Further data illustrated that 75 percent of the members of both groups (MIS and functional) viewed work experience as even more important than graduate study.

Table 3: Master's Degree Essential for Career Advancement by Functional Status

	MIS		FUNCTIONAL		TOTAL	
	#	%	#	%	#	%
1 Strongly Agree	10	38.5	7	29.2	17	34.0
2 Agree	7	26.9	7	29.2	14	28.0
3 Neither	3	11.5	5	20.8	8	16.0
4 Disagree	6	23.1	3	12.5	9	18.0
5 Strongly Disagree	0	0.0	2	8.3	2	4.0
TOTAL	26	100	24	100	50	100

Motivations for Graduate Study

With regard to the question of motivation, about two-thirds of the functional area professionals and slightly over half of the MIS professionals cited acquiring a better business background as a primary reason for pursuing a Master's degree. See Table 4: Table 4: Primary Reason for Master's Degree-Business Background By Functional Status

	MIS		FUNCTIONAL		TOTAL	
	#	%	#	%	#	%
1 Strongly Agree	5	19.2	4	16.7	9	18.0
2 Agree	9	34.6	12	50.0	21	42.0
3 Neither	8	30.8	6	25.0	14	28.0
4 Disagree	4	15.4	1	4.2	5	10.0
5 Strongly Disagree	0	0.0	1	4.2	1	2.0
TOTAL	26	100	24	100	50	100
MEAN	2.42		2.29		2.36	

Once technical professionals have gained three to five years' experience in programming and analysis, they may aspire to positions as project managers and supervisors. They need a better business background in order to understand users' requirements and in order to manage people, time, and resources to accomplish design projects.

Technical knowledge is a lesser reason for pursuing graduate study, although almost half of the functional area managers pointed to the need to acquire technical

knowledge. Managers with an accounting, marketing, or sales background may want to improve their technical knowledge when they find themselves in the position of making decisions on local area networks, software packages, and microcomputers. See Table 5.

Table 5: Primary Reason for Master's Degree-Technical Knowledge By Functional Status

	MIS		FUNCTIONAL		TOTAL	
	#	%	#	%	#	%
1 Strongly Agree	2	7.7	3	12.5	5	10.0
2 Agree	6	23.1	8	33.3	14	28.0
3 Neither	5	19.2	3	12.5	8	16.0
4 Disagree	10	38.5	9	37.5	19	38.0
5 Strongly Disagree	3	11.5	1	4.2	4	8.0
TOTAL	26	100	24	100	50	100
MEAN	3.23		2.88		3.06	

As you can see from these data, the MIS professionals placed less emphasis on technical skills. Only 30 percent noted that technical skills are a motivation for a Master's degree, and about half "disagreed" that technical knowledge was a reason for graduate study. This can be explained by the fact that many MIS professionals already have a technical background and wish to acquire a business background to assume managerial responsibilities in the future.

Because of the emphasis on attaining a "business" background, as opposed to a "technical" background, the MIS professionals preferred an MBA degree with an MIS specialization to an M.S. in Management Information Systems or a general MBA degree. While about one-third of the functional area professionals preferred an MBA with an MIS specialization, most of them sought a general MBA degree. See Table 6:

Table 6: Most Beneficial Degree by Functional Status

	MIS		FUNCTIONAL		TOTAL	
	#	%	#	%	#	%
MS MIS	6	23.1	1	4.2	7	14.0
MBA General	9	34.6	16	66.7	25	50.0
MBA Spec. MIS	11	42.3	7	29.2	18	36.0
TOTAL	26	100	24	100	50	100

It seems that some functional managers want to acquire technical knowledge in the context of an MBA degree, while others prefer a generalist degree. MIS professionals definitely sought the MBA degree as a strategy for gaining managerial skills.

Competencies within a Master's Program

The respondents were asked to identify and to rank managerial skills, technical skills, and management of MIS skills which they wanted to acquire as part of a Master's program.

In terms of managerial skills, the top ranked competency was "managing and motivating personnel." While the MIS professionals ranked "project budgeting and control" second, the functional managers felt that "goal setting and strategy development" was second in importance.

Table 7: Ranking of Managerial Skills By Functional Status

	MIS		FUNCTIONAL		TOTAL	
	Rank	#	Rank	#	Rank	#
Goal Setting & Strategy Dev.	3	38	2	46	2	84
Project Budgeting & Control	2	41	4	21	3	62
Making Org. & Human Resource Decision	4	23	3	27	4	50
Managing/Motivating Personnel	1	54	1	50	1	104

The first-ranked technical skill was "information systems design," followed by "advanced development tools." This was true for both the MIS professionals and the functional managers.

Table 8: Ranking of Technical Skills By Functional Status

	MIS		FUNCTIONAL		TOTAL	
	Rank	#	Rank	#	Rank	#
Advanced Development Tools	2	44	2	33	2	77
Programming	4	15	4	13	4	28
Technical Knowledge Hardware/Software	3	30	3	29	3	59
Information System Design	1	61	1	51	1	112

As you can see from Table 8, the technical skills, including knowledge of hardware/software and programming, were considered less important. In fact, programming was of relatively minor importance. This is consistent with the expressed need for a "managerial" rather than "technical" focus within an advanced degree.

In terms of the competencies associated with management of MIS, the top-ranked item was "strategic management," or the process of identifying information technology plans which support organizational plans. This was foremost in the minds of the MIS professionals. The second-ranked competency was "managing the systems development life cycle," and the third was "evaluating alternative development methods." See Table 9.

Table 9: Ranking of Management of MIS Skills By Functional Status

	MIS		FUNCTIONAL		TOTAL	
	Rank	#	Rank	#	Rank	#
Managing Systems Dev. Life Cycle	2	39	2	31	2	70
Organizing MIS	4	28	4	17	4	45
Evaluating Alternative Dev. Methods	3	29	3	23	3	52
Strategic Management for MIS	1	60	1	37	1	97

These issues are clearly in the minds of both functional managers and MIS professionals, because of the overwhelming need to improve systems development productivity.

Important Factors in a Master's Program

Finally, the respondents were asked to rank-order a number of factors which are associated with program quality. The most important factor was "quality of program faculty," followed by the opportunity to gain "hands-on experience." "AACSB accreditation" and "convenience" were lesser concerns.

Table 13: Ranking of "Ideal" Master's Program Factors By Functional Status

	MIS		FUNCTIONAL		TOTAL	
	Rank	#	Rank	#	Rank	#
Quality of Program Faculty	1	52	1	54	1	106
Hands-On Experience	1	52	2	47	2	99
AACSB Accreditation	3	28	3	26	3	54
Convenience of Location	4	24	4	17	4	41

CURRICULUM RECOMMENDATIONS

The Master's Market Research study has led to the development of two directions for graduate study. The first of these directions is the design of an MBA specialization emphasizing managerial competencies. The second curriculum initiative is the design of a joint Master of Science in Computing and Information Systems with the Department of Computer Science.

The MBA specialization is designed to prepare students in the management of new information technology and the management of information systems design projects. The four courses in the MBA specialization include:

MIS 570: Information Systems Analysis

MIS 572: Information Systems Design

MIS 564: Database Design

MIS 540: The Management of Information Systems Development

The Master of Science in Computing and Information Systems

The second option for graduate study in information systems is a new program entitled the Master of Science in Computing and Information Systems. This program is a joint program of the Department of Management Information Systems within the School of Business and the Department of Computer Science within the School of Science. The program

provides both a "technical" and a "managerial" focus.

The Program of Study

The program of study consists of a Computing and Information Systems core which provides a foundation in both technical concepts and systems design:

The six-course core includes:

MIS 564: Database Management Techniques
MIS 570: Structured Analysis Techniques
MIS 572: Structured Design Techniques
CS 514: Operating Systems
CS 516: Computer Architecture
CS 520: Data Communications and Networks

In addition to the core, the program provides students with an opportunity to take four electives from one of two groups. One of these groups emphasizes software engineering techniques, and the other focuses on computing technologies. Examples of courses from each of these groups include:

Software Engineering Elective Group:

MIS 588a: Software Engineering
MIS 588b: Advanced Database and Information Engineering
MIS 540: Management of Information Systems Development
CS 425: Software Project Development
CS 537: Expert Systems

Computer Technology Elective Group:

CS 416: High Performance Computer Systems
CS 438: Introduction to Artificial Intelligence
CS 482: Computer Graphics

The M.S. in Computing and Information Systems will have a culminating project which will involve the analysis, design, and implementation of a system using technologies and methodologies which are covered in the program.

CONCLUSIONS

The market research study for the Master's program in Management Information Systems provided a number of effective insights. The department faculty developed an MIS specialization within the MBA degree in response to the increasing interest of functional area managers in information systems development.

In contrast, the plans for the new M.S. in Computing in Information Systems address a different need. This program provides a technical and a managerial focus for those who are interested in a comprehensive set of courses in computing technology and its applications.

REFERENCES

Boland, R. J., "The Process and Product of System Design," Management Science, Vol 24, No. 9, May, 1978, pp. 887-898.

Cheney, Paul H. and Lyons, Norman, R., "Information Systems Skill Requirements: A Survey," MIS Quarterly, Vol. 4, No. 1, March 1980, pp. 35-43.

Green, Gary I., "Perceived Importance of Systems' Analysts' Job Skills, Roles, and Non-Salary Incentives," MIS Quarterly, June, 1989, pp. 115-133.

Kaiser, K. and Srinivasan, A., "User-Analyst Differences: An Empirical Investigation of Attitudes Related to Systems Development," Academy of Management Journal, V. 25, No. 3, September, 1982, pp. 630-646.

WHAT THEY DIDN'T TEACH YOU IN COLLEGE:

ETHICS TRAINING IN INFORMATION SYSTEMS EDUCATION

Mark W. Smith, Ed.D.
Department of Computer Technology
Purdue University

ABSTRACT

As a nation, we face a crisis in ethics. Not teaching professional ethics in your information systems classes can only make it worse. Daily IS professionals face ethical dilemmas. As IS educators it is our responsibility to help train our students to make sound ethical decisions. We can teach professional ethics in our IS courses! This paper attempts to guide the IS faculty member in the exploration of the range of help available to meet this challenge. Also, in the presentation, an ethics quiz will be administered demonstrating how easily we can motivate our students to explore the ethical conflicts in which some day they may find themselves. Remember, training students on how to deal with ethical dilemmas prepares them to be IS professionals as well as fulfilling your responsibility as a professional IS educator.

SOCIAL ISSUES OF INFORMATION TECHNOLOGY: PARADIGMATIC CHANGES IN THE INFORMATION AGE

Roy M. Dejoie (University of Oklahoma) and George C. Fowler (Texas A&M University)

ABSTRACT

Computer technology has become a classic and expressive mainstay of our society quicker than we've been able to assess its impacts. It is imperative that we assess the possible changes that computer technology may bring and has already brought to our society. This paper presents some of the research that has addressed the social issues concerning computer technology. A parallel between the computer and the automobile, regarding their societal impacts, is also presented. Paper references available upon request.

INTRODUCTION

The social aspects of most new technologies are often overlooked until the technology has become rooted in society's fabric. Technologies, in general, are usually viewed by their "bottom line." Often a technology's use starts off as "classic" (where technology is used to do activities better or more easily or more effectively than they are being done without the technology) [2]. When a technology is used classically, it is easy to overlook any broader effects that it might extend beyond its classical use. Technologies can also have an "expressive" use (where the technology is used to do things that have been previously considered impossible) [2]. The computer started out in classic use, but its meteoric progression to expressive use has surpassed all other technologies.

Because computer technology has become a classic and expressive mainstay of our society quicker than we've been able to assess its impacts, it is imperative that we assess the possible changes that computer technology may bring and has already brought to our society. This paper will present some of the research that has addressed the social issues concerning computer technology, but first a prophetic parallel between the computer and the automobile will be presented.

A CONTINUING LESSON FROM HISTORY: THE AUTOMOBILE

When the automobile was first introduced it is unlikely that anyone could have foreseen the social aspects of this technology. Also the very advent of most technologies tend to conjure up the dreams and ideologies possible. Glen Jeansonne [3] states:

> The American automobile has traveled the whole circuit from hero to villain. Once enshrined as a liberating and democratizing agent, it is now condemned as a major cause of pollution and congestion.

As a whole, it is highly unlikely that many of us can truly visualize the total impact of a new technology during its advent and early growth. This is particularly true of the social aspects. Once again, using the automobile as an example, the following quote illustrates this point:

> Not a phase of American life was untouched by the automobile. Americans no longer measured distance in miles, but in minutes. Isolated wasteland became choice property when a major highway cut through it. The automobile made elopements easier, increased the incomes of parsons who specialized in quick matrimonials, and swelled the duties and fees of village constables. It made bootlegging profitable and prohibition impractical. It enriched the American vocabulary with such words as "flivver", "skid", and "jaywalker." It captured the hearts, imaginations, and pocketbooks of Americans [15].

This was written in 1923. Imagine the surprise of the author if he had gotten a glimpse of what the automobile would come to symbolize in the 1990's. It is treated as a status symbol while at the same time it is considered a necessity for modern living. It also takes on the role of private "miniature home away from home." How many times do we notice people singing, eating, fixing their hair, putting on make-up, "making out" in the backseat or even picking their nose (there are not too many more private things in life)? The sense of privacy is surely one of the aspects that was probably not envisioned during the advent of the automobile.

Beyond the enhanced degree of privacy, the automobile moved quickly from classic use to expressive use [2]. Hardison [2, p. 237] states

> A truly new technology refuses to stay classic. Even if it was first created for a classic function, it eventually becomes expressive and reshapes the function. ... The success of the automobile created so many new

> conditions that society had to be reshaped to accommodate them. In spite of the best of early inventions, within a few years after its commercial introduction the automobile ceased to be classic and became expressive.

Toffler [22, p. 64] contends, "It was the automotive industry that first succeeded in destroying the traditional notion that a major purchase had to be a permanent commitment." The automobile set the pace for "disposable" possessions which in turn has fueled our "disposable society." In 1970, the average car owner kept their car for three and a half years [22]. It has become easier to buy a new car than to repair and maintain the old one (it's interesting that most standard factory warranties are for three years). We live in a society where food is processed in a couple minutes, picked up at "drive thru" windows, consumed while traveling on to our next destination and upon arrival the "dinnerware" (a paper wrapper, cardboard container, wax paper cup and a napkin) is disposed of in a brown paper bag. Information technology mimics this same disposable atmosphere. We erase the contents of a diskette in the blink of an eye. Data can be discarded nonchalantly in many cases as the time and effort to recalculate or restore the data is minimal. The low cost of storage media makes it easier to discard a damaged disk rather than repair it (assuming that the data on it can be restored).

In another parallel, Muller [13] states that the automobile stimulated the emigration to the suburbs and, in doing so, changed our cities into "undefined metropolitan areas" [13, p. 99]. Muller further shares

> As it created such other novelties as supermarkets, shopping centers, and drive-ins, suburbanites grew absolutely dependent on it. It took fathers to work, mothers to the store, youngsters to school. Teenagers used it to make love in or to go on joy-rides. And from the beginning it had brought joy to the hearts of Americans, an exhilarating sense of freedom and power as they drove, and of possible social mobility too.

Computer technology is fast changing our social orientation. Computers allow workers to work at home, thus redefining the role of the office and the home simultaneously. Information services such as Prodigy allow families to "shop" from their home. Libraries and other information sources are easily accessible via "dial-in" lines. Regarding the similar parallel in population movement, Naisbitt and Aburdene [14, p. 307] contend that "... if cities did not exist, it now would not be necessary to invent them." They assert that the new electronic "cities" will be populated by people that are not location-dependent or location-bound [14]. "We do not have to cluster together in cities or suburbs to get our work done as we did during the industrial era" [14, pp. 306-307]. Not only will information technology have an influence on "how" we live, but perhaps on "where we live".

A final note concerning emerging technologies is that technologies can sometimes undue the paradigmatic effects brought on by a previous technology. Computer technology is already starting to supplant grocery stores and various other services that were by-products of the automobile's reign. As mentioned previously, information services are already allowing consumers to shop at home by allowing them to place their grocery order via their computer and having the grocery store deliver the items. Communications technology has also allowed consumers to sit at home, while viewing television, and shop for household items from rings to clothing to furniture. One nostalgic supplanting is evident in the closing of drive-in theaters due to new forms of emerging communications technology (cable, satellite, video cassettes, etc.). Last Saturday (fittingly a special date, February 29, 1992), the I-45 Drive-In, the last drive-in theater in Houston, showed its last movie before silencing its screens forever and bringing to an end an era of entertainment ushered in by the automobile.

These effects are likely to be similar to the way that computer technology may also impact our social lives. Just as with the automobile, there are bound to be those social aspects that will be missed while we are concentrating on the "bottom line." A suggestion that may be made is, given the social impact of the automobile, it might be wise if we also focus on the social aspects of computer technology instead of only focusing on the "bottom line".

SOCIAL ISSUES OF INFORMATION TECHNOLOGY

Research into the social aspects of computer technology is relatively new. Additionally, this area is currently lacking in empirical studies. The majority of the research actually consists of papers positing the possibilities and areas of social impact that computer technology is likely to affect. Table 1 (available upon request) summarizes the research done regarding the social aspects of computer technology.

Earlier research began in the late 1960's when Sackman [21] developed a general theory and philosophy of man-machine systems. Little work was pursued for the next decade, until several events in the 1980's occurred. Throughout the 1980's, Computer Professionals for Social Responsibility (CPSR), provided a focal organization for those interested in the social and ethical impacts or computer technology. Douglas Johnson [6] outlined the characteristics for a computer society ethic. Deborah Johnson [4] [5] identified ethical issues regarding computer use including the equal access to computing and computing expertise. Richard Mason [11]

identified privacy, accuracy, property, and access as four ethical issues that warranted special consideration in the information age. Jan Zimmerman [24] examined the influence of information technology on females and envisioned a greater discrepancy between males and females as information systems begin to limit or eliminate the need traditionally female dominated jobs including bank tellers, telephone operators, librarians and retail sales clerks. LaChat [9] posited questions of the ethical issues concerning artificial intelligence research and experimentation in terms of medical experimentation. M. Mitchell Waldrop [23] addressed the relationship between man and artificially intelligent machines. Kling and Iacono [7] discussed the role of computerization in reforming society and the world. Gandy [1] asserted that advantages provided by advanced electronic technologies in the workplace, marketplace and government are fueling the trend toward automatic methods of surveillance of individuals that the United States legal system cannot control or keep pace with. Krendl et al.'s [8] three-year study of middle- and high-school students found that girls were less interested in computers and less confident in their computer skills (compared to boys) even when their experience with computer technology was comparable with boys. Linowes [10] addressed a number of issues associated with the Information Age including privacy, education, national databanks and politics. Paradice and Dejoie [16] presented research that suggests that the presence of computer-based information systems in an ethical dilemma might lead to more socially-oriented decision-making processes in resolving the dilemma.

Investigations in this area are relatively recent, but a movement in understanding the social aspects of computer technology is gaining momentum. The area of human factors is a related, growing area. A better understanding of prior studies helps us comprehend the origin of social aspects study and the direction it is headed.

SUMMARY OF RESEARCH IN SOCIAL ASPECTS

Sackman's development of a general theory and philosophy of man-machine systems included the tenet of "humanistic automation." Sackman stated that computer systems existed to meet human needs and serve social well-being. Given Sackman's theory, the process of automation was not required to assume the role so drudgingly depicted in Fritz Lang's masterpiece, Metropolis. Sackman also observed that ethics and values could be treated as research hypotheses and could therefore be examined scientifically.

Douglas Johnson [6] addressed ethical issues involving the proliferation of computer-based information. He focused primarily on the need for modification of existing rules governing human behavior. His work outlined the characteristics that an "ethic for a computer society" should exhibit.

Deborah Johnson [4]([5] presented the idea that while computers have not currently caused a fundamental change in our social institutions they will "pose new versions of standard moral problems and moral dilemmas, exacerbating the old problem, and forcing us to apply ordinary moral norms in unchartered realms" [4, p. 1)]. She suggested that the computer transforms the context in which professional-ethical issues arise. Additionally, Johnson addressed the concerns of equal access to computing and computing expertise, including the role that computers will play in the distribution of rights and benefits in the future information society.

Richard Mason [11] addressed ethical behavior in terms of privacy, accuracy, property, and accessibility. Mason states there exist certain data items which individuals have a right (1) to keep private, (2) to ensure are accurate, (3) to own, and (4) to access. He noted that "information forms the intellectual capital from which human beings craft their lives and secure dignity" (p. 5). However, information technology may be misused in each of these four critical areas, reducing an individual's intellectual capital.

Michael LaChat [9] approached the topic of artificial intelligence research from a philosophical and medical standpoint. He constructed the premise of a "personal" AI entity, one with consciousness and awareness. After constructing this premise he then questioned the morality of AI experimentation in terms of medical experimentation ethics. He also questioned the possibility and capacity of such an entity making moral decisions.

Zimmerman [24] examined the impact of information technology on females. She asserts that the growing movement toward computerization will have a larger negative impact on females. She points out that a majority of the jobs that will become obsolete because of the Information Age are jobs that are predominantly held by women (librarians - 85% women, retail sales clerks - 60% women, billing clerks and cashiers - 85% women, telephone operators - 92% women and bank tellers - 94% women). On the other side of the coin, traditionally male dominated jobs including shipping clerks, electronic technicians, TV repair and installation technicians, computer repair and installation technicians and home appliance repair and installation technicians will all continue to proliferate. She also examines the relationship between the family movement and telecommunications technology which may threaten women's advancement and progression in corporate settings.

M. Mitchell Waldrop [23] examined the future relationship of man and machine. In his book, Man-Made Minds: The Promise of Artificial Intelligence, Waldrop [23] envisioned the possible future of computer technology and drew parallels between the invention of the printing press in Renaissance times and the advent of computer technology in our present times. He also addressed the trend of automation and gave possible alternative workstyles and lifestyles by

varying automation scenarios. He confronted the degree of responsibility that should be given to computers in our society. He posited the possibility of artificially intelligent entities that interacted in man's society and the degree to which these machines should be given various responsibilities. His theory of machine ethics created rules to govern moral and ethical behavior of AI entities among their human counterparts.

Kling and Iacono [7] examined the computer movements and the possibility of their influence on societal reforms and world reforms. Areas that are the main thrusts of the movements include urban information systems (systems that process information about the activities of individuals in local government jurisdictions, the services they receive and the internal operations of local governments), artificial intelligence, computer-based education, office automation and personal computing. While the computer movements are considered revolutionary, Kling and Iacono suspect that the revolution will be a fairly conservative one and that it is likely to reinforce "the patterns of an elite-dominated, stratified society."

Gandy [1] contended that the recent advanced electronic technologies provide advantages in the workplace, marketplace and government which fuel the trend toward automatic methods of surveillance. He further contends that the United States legal system cannot control this proliferation of surveillance. He states that privacy legislation has done very little to preserve or extend the rights and freedoms of individuals. Additionally, the laws that do exist focus more on government activities than on corporate activities (which Gandy feels has a far more extensive reach than government activities).

Krendl et al. [8] conducted a three year study of middle- and high-school students. They found that girls were less interested in computers and less confident in their computer skills than boys even when their experience with computer technology was comparable. They also found that the differences did not diminish over time as the children became more experienced with the computer technology. An interesting result that was found was that confidence in the ability to use information technology decreased as the students gained experience with the technology and that this was more pronounced in girls than in boys.

Linowes [10], in a speech delivered at the White House Conference on Libraries and Information Services, described the influences of information technology on topics such as education, privacy, politics and national databanks. He also expressed the need for libraries to focus on policies and activities that would help make the Information Age transition smooth and beneficial to all. Along with libraries, he also envisioned mayors and governors, local schools and museums acting as catalysts, stimulants and to bring together industry, school and home to promote adoption of various programs that would be appropriate for the ongoing Information Age.

Paradice and Dejoie [16] researched the influence of information systems on ethical decision-making processes. They used the Defining Issues Test (DIT), developed by James Rest [17] [18] [19] [20] and a modified version of the Defining Issues Test (which included a computer-based information system). The DIT presents moral dilemmas to subjects followed by a list of issues that a person might consider in resolving the dilemma. Paradice and Dejoie found people are more likely to consider more socially-oriented issues when attempting to solve an ethical dilemma when a computer is present.

A recently formed organization, Computer Professionals for Social Responsibility (CPSR), emphasizes the ethical responsibilities of computer industry professionals. CPSR's main focuses have been privacy rights and computers and the military. They have, however, also addressed many other social and ethical issues including responsibility for faulty computer programming, computers and South Africa, viruses, voting, and education. The organization is comprised of many noteworthy individuals and has made great strides in introducing legislation to protect individuals' rights as well as opposing legislation it believes threatens those rights.

FUTURE RESEARCH ISSUES IN SOCIAL ASPECTS

Many future research areas regarding the social aspects of computer technology exist. In light of Douglas Johnson's work regarding the characteristics of a computer society ethic, one might compare the evolving computer society ethic to the society ethic that evolved during the Industrial Revolution. Deborah Johnson's work could be extended with comparative studies of the disparity between socio-economic classes, education levels, races, genders, and so forth with regard to computing access.

Richard Mason's research provides a basis for further investigations into the areas of privacy, accuracy, property, and access. Research concerning the ownership of information, national databases, responsibility, information maintenance, copyright laws, ownership, liability, and equal access to information are specifically needed to ensure that individuals' rights are considered in the information age.

More speculative pieces, like Michael LaChat's article, are needed to provide interdisciplinary focus. The area of medicine has already experienced moral and ethical conflicts, for example in genetic engineering. Parallels between the moral and ethical challenges in the medical field and computer field should be investigated.

M. Mitchell Waldrop's theory of machine ethics could be explored more. If conscious AI entities are possible there must be some means for conveying and instilling ethics and morality into their artificially intelligent minds. Even if nothing comes of the study of machine ethics, Waldrop asserts that the knowledge gained will be more than worth the effort.

Krendl et al.'s [8] and Zimmerman's [24] research suggests that some future research should focus on the overall impact of information technology on women. Women are a majority of the United States population; however, they are likely to be negatively affected by information technology. Investigations of displaced employees, due to obsolescence of many jobs because of their replacement through information technology advancements, should study the "focused obsolescence" of jobs, in which a particular part of the population (for instance women or other minority groups) is overly affected by shifts due to obsolescence of their jobs due to information technology. Job relocation and retraining will need to be given serious priority.

Gandy's research [1] suggests a future research endeavor in the role of surveillance and individual action. Just as the Hawthorne studies altered employee behavior because of the mere fact that the subjects knew they were being watched, so too might surveillance practices lead to the altering of employee behavior. Unlike the Hawthorne experiments though, it is possible that the surveillance movement may have further reaching effects. Miller [12, p. 105] states,

> It doesn't matter whether there is a Big Brother on the television screen. It doesn't matter at all whether somebody is, in fact, watching you. The only thing that does matter is that you <u>think</u> that the is somebody watching you. You'll do the rest to yourself. We're human beings; we react. We don't even know we're reacting, that we are modifying our behavior.

In light of this, research into the <u>perceptions</u> of surveillance invasions should be conducted to determine the extent to which actual and perceived surveillance influence employee and private citizen behavior.

A longitudinal study is needed to determine whether socially-oriented decision-making processes continue as computer technology is assimilated into society. Also, researchers must determine whether the introduction of any new technology produces the same type of socially-oriented decision-making processes.

CONCLUSION

As stated earlier, many social aspects of computer technology are discussion-oriented. As the technology is assimilated into society these issues will become more empirically measurable. Until that time, these issues must not be shunned just because they do not easily lend themselves to empirical testing. Through the discussions of the possibilities today we may be able to avoid problems tomorrow.

Group Differences and Ethical Issues in Information Systems

by
George C. Fowler
Department of Business Analysis and Research
College of Business and Graduate School of Business
Texas A&M University
College Station, Texas 77843
and
Roy Dejoie
Division of Management
College of Business
University of Oklahoma
Norman, Oklahoma 73019

Research into the interpretation of ethical dilemmas by different groups of individuals is relatively new. Studies into group differences look at such questions as "Do Accountants look at moral issues differently than MIS managers?" This paper reviews the literature in the area of group differences. The literature goes back to Weinberg's 1971 book in which he noted for the first time that data processing managers faced ethical dilemmas. This review includes Parker's work in 1979 which used scenarios to pose ethical dilemmas a variety of groups of individuals to explore group differences. Some of the more current work includes the research by Shim and Taylor in which they explored the issue of software piracy among groups of professors and managers. Paradice and Dejoie's research used scenarios to explore differences in MIS and Computer Science students.

Much knowledge can be gained about ethical issues and information systems by studying the interpretation of ethical dilemmas by groups of individuals. While the area of research is only 20 years old, much recent interest can be seen. This paper hopes to present a summary of where the research has been and to propose future research in the area of group differences.

Computer Threats...An Analysis of Emerging Trends

Dr. Karen Forcht
Dr. Joan Pierson
Department of Information
and Decision Sciences
College of Business
Showker Hall
James Madison University
Harrisonburg, VA 22807
703-568-3064/3057/3055

ABSTRACT

This paper outlines the current threats to computer systems. The computer environment has been emerging as the sight of computer abuse. A definition of computer abuse is thoroughly discussed, as well as the different types of computer criminals. The motivations behind the various computer crimes is analyzed, along with preventative measures that can be taken.

Conclusion:

After analyzing who the computer criminal is and what his or her motives are, some basic conclusions can be made. It is obvious that the problem of computer crime is a reality and has been steadily increasing over the past years. In order to determine how to approach this dilemma and decrease the number of crimes committed, an organization must identify the culprit.

PRELIMINARY FINDINGS OF A STUDY ON THE AFFECT OF CLASS SIZE ON TEST SCORES IN AN INTRODUCTORY COMPUTER COURSE

Jo-Mae B. Maris and Evangeline L. Jacobs
Computer Information Systems, Northern Arizona University

ABSTRACT

One option for "doing more with less" in this era of diminishing university funding is to increase class size. This study considers the relative effectiveness of small versus large classes in the introduction to computer information systems course at a state university. The common exam scores for eight small sections (about 50 students each) and five large sections (about 150 students each) are compared. As might be expected, the smaller sections performed significantly better than the larger sections. A preliminary student survey was used to measure variations in the composition of sections. The factors considered included highest level of math completed, classification (i.e., freshman, sophomore, junior, senior), and instructor. Class size was one of the significant factors for explaining variation in performance among students.

INTRODUCTION

Purpose

The purpose of this study is to determine whether reducing the class size from 150 to 50 improves students' learning as measured by test scores.

Problem

Providing adequate education in high enrollment courses is a problem faced by many curriculum areas. Balancing of faculty resources and course offerings is affected by the size of the sections offered. Fewer faculty are required to teach a course with high enrollments if class sizes are increased.

At a state university, the introductory course in computer information systems has enrollments of approximately 800 students per semester. If traditional sections of thirty to thirty-five students were used, twenty-four sections would be necessary to accommodate eight-hundred students. Using larger class sizes reduces the number of sections needed to four or five. To cover the twenty-four traditional sections would require six faculty, each teaching four sections of thirty to thirty-five students. The large sections, however, could be covered by as few as three faculty members with two faculty members doing the lecturing and an administrative coordinator. The faculty estimates for both the traditional sections and large sections assume separate computer applications laboratories (similar to science laboratories). Faculty requirements for the laboratories are excluded since they remain constant.

Thus, the question is do small sections result in improved performance that would justify the use of additional faculty resources.

Method of Investigation

The data used in this preliminary report were gathered during the Fall 1991 and Spring 1992 semesters at a state university. During the Fall 1991 semester, six sections of the introductory computer information systems course were offered. Three sections had a capacity of 160 students each, and the other three sections had a

capacity of 50 students each. During the Spring 1992 semester, seven sections were offered. The three 50 student sections filled to capacity. Two of the four 150 student sections filled to capacity. The other two 150 student sections ended with enrollments of about 50 students. For the purpose of this study the under-filled sections will be considered small sections.

Full-time faculty taught the sections both semesters. No adjunct staff taught any of the sections. The composition of the sections is discussed in the "Analysis of Data" portion of this paper.

The data consist of preliminary survey results and test scores. The students in each section were required to complete a survey at the beginning of the course. The survey questions addressed student characteristics, such as classification, major, math classes completed, and previous computer experience.

The students took four tests during the semester. The tests consisted of sixty-five multiple choice questions. Each question was worth two points. The maximum score on a test was 130 points. The first three tests were the same for both semesters. The fourth test was modified for the Spring semester. The same tests were used for all sections each semester.

Scope

Little has been published concerning the affect of class size on performance of university students. The ERIC[1] database contains no references for class size in higher education computer courses. Lewis[2] is the only reference in ERIC that addresses a comparison of small and large class sizes. Lewis' comparison is based on preference rather than performance. Most of the references in the ERIC database concern how to teach large sections, rather than the comparative performance of large versus small sections. The current study concentrates on performance of students on identical tests.

Lewis reports that most students and teachers prefer class size to be under fifty persons. The small sections offered in the introductory computer information systems course were at the upper-bound of this preferred class size. The sample sizes of the small sections range from thirty-five to fifty-five students. The sample sizes of the large sections range from 122 to 146 students. Consequently, this study concerns sections of about 135 students and about 43 students.

The small and large sections will be evaluated with respect to mean performance on tests. In addition an analysis of variance was performed to investigate the contribution of section size to explaining the variance in the mean performance on tests.

ANALYSIS OF DATA

In the analysis of the data, the primary concern is whether future small sections are likely to perform better than large sections. To make inferences requires that the data used represent the group about which the inference is made. To the best of our knowledge, the composition of the introductory computer information systems course enrollments have been fairly consistent. Therefore, the assumption is made that the Fall 1991 and Spring 1992 students are a fair representation of future students.

The data were purged on two criteria. First, observations were eliminated if no survey was completed although test scores existed. Second, observations with missing test scores were eliminated. The resulting data set contained 1019 observations. The data set represents seven small sections (35..55 students) and five large sections (122..146 students).

---

[1] Education Resource Information Clearinghouse is a CD-ROM database that combines the Research Index for Education (RIE) and the Current Index for Journals in Education (CIJE) indexes.

[2] Lewis, Karron G., "The Large Class Analysis Project (Final Report)," University of Texas at Austin, June 1982.

The composition of the sections shows some variations. The small sections have a statistically significant larger proportion of upper classmen (see Table 1). However, the sections seem to have similar math levels (see Table 2). These are the two class composition characteristics used in the analysis of variance.

Table 3 summarizes the mean and dispersion statistics for the first test and the performance score (average of tests two, three, and four). The "Mean" is the arithmetic mean,

$$\bar{x} = (\sum_{j=1}^{n} x_j) / n. \quad (1)$$

Equation 1 is used for each grouping. In Equation 1, $\bar{x}$ represents the mean; n indicates the sample size, and x_j denotes the test score of the j_th student in either a large section or the test score of the j_th student in a small section. "Std.Dev." is the standard deviation (square root of the variance). The variance was computed using the sample formula,

$$s^2 = [\sum x_j^2 - (\sum x_j)^2/n] / (n-1). \quad (2)$$

The "n" row in Table 3 contains the sample sizes. The F-statistic, shown in the row labelled "F," was computed using

$$F_{performance} = s^2_{large} / s^2_{small}. \quad (3)$$

The row labelled "t' score" in Table 3 is the t-statistic with unequal variances. The equation used to compute t' was

$$t' = \frac{(\bar{x}_{large} - \bar{x}_{small})}{SQRT(s^2_{large}/n_{large} + s^2_{small}/n_{small})} \quad (4)$$

To determine whether the small sections would perform better, the average of Test 2, Test 3, and Test 4 (hereafter called performance score) for each student were analyzed. The following hypotheses were used.

$$H_0: \mu_{large} \geq \mu_{small} \quad (5)$$
$$H_a: \mu_{large} < \mu_{small} \quad (6)$$

In Equations 5 and 6, "μ_{large}" indicates population mean of performance scores for large sections. "μ_{small}" indicates the population mean of performance scores for small sections. The test statistic is t' (Equation 4). The rule for rejection is to reject the null hypothesis, if t' is less than $-t_{\alpha,df}$. "df" was determined by

$$df = \frac{(s^2_{lg}/n_{lg} + s^2_{sm}/n_{sm})}{(s^2_{lg}/n_{lg})^2/(n_{lg}-1) + (s^2_{sm}/n_{sm})^2/(n_{sm}-1)} \quad (7)$$

SAS output was used in testing H_0. An α equal 0.05 was chosen. The SAS-generated degrees of freedom were 737.4. Since t-tables are not readily available for degrees of freedom greater than 120, the probability of a greater t' reported by SAS was used instead of a critical t' value. Thus, H_0 was rejected if the probability of a greater t' was less than 0.05. The t' reported by SAS was -6.0923 with a probability of a greater t' of 0.0001. Since 0.0001 is less than 0.05, the null hypothesis was rejected. Based on this t-test, small sections have significantly better performance scores than large sections. The t-test for unequal variances, t', was used because no assumption is made concerning the variances.

Although the t-test indicates a significant difference between the performance scores of small and large sections, the presence of a higher proportion of upper classmen in small sections complicates the analysis. To take into account the difference in composition of the small and large sections, an analysis of variance was used. The analysis of variance allows consideration of several factors to explain the variation in performance scores.

The factors (or independent variables) used to explain the variation in performance scores were the first test score, students' self-reported math level, students' self-reported classification, the instructor codes for the sections, and section size code. The performance scores were the

dependent variable. The factors used to measure student ability were the first test score, math level, and classification. The factor of interest is small versus large sections. Test 1 is an interval variable with a range of 0 to 130. The scores on the first test are not significantly different between small and large sections, see Table 3. Math level has four values: (1) high school algebra, (2) intermediate algebra, (3) college algebra, and (4) higher than college algebra. Classification has six values: (1) freshman, (2) sophomore, (3) junior, (4) senior, (5) graduate, and (6) other. The instructor code has four values (3 to 6). A main effects model was used.

The results obtained from SAS are shown in Table 4. This model explains a little over half of the variation in the performance scores (R-Squared=0.56). The model is better than the mean of the performance scores at explaining their variation (probability of a greater F equals 0.0001). As shown in Table 4, both section size and classification are important in the explanation of the variance, but section size appears to be a stronger factor. The small section's probability of a greater F (0.0001) is stronger than the classification's probability of a greater F (0.0676).

CONCLUSIONS

Even based on these preliminary results, one can conclude that small sections perform better than large sections (t-test on performance scores). The only factor clouding an unconditional conclusion of smaller sections performing better than larger sections was the composition of the sections. The small sections had more upper classmen than large sections. However, the analysis of variance indicates that the section size is a stronger factor than the classification. Thus, our conclusion is that small sections do improve the performance of students in introductory computer information systems courses.

RECOMMENDATIONS

If one of the university's top priorities is to improve the education offered to students, then small section should be used. However, if the university's top priority is to reduce faculty, no recommendation made here will be relevant.

TABLE 1. Proportion of Upper Classmen

Section Size	Freshman Sophomore	Other	Total (n)
Small			
Freq	203	140	343
Proportion	.5918	.4082	
Expected	261	82	
Large			
Freq	573	103	676
Proportion	.8476	.1524	
Expected	515	161	
Total	776	243	1019
P_0	.7615	.2385	

H_0: $P_{lg} = .7615$ H_a: $P_{lg} \neq .7615$

Reject H_0, since

$|Z_{calc} = 5.25| > Z_{\alpha/2} = 1.645$

$$Z_{calc} = \frac{P_{lg} - P_0}{SQRT(P_0(1-P_0)/n))} = \frac{.8476 - .7615}{SQRT(.7615(.2385)/676)}$$

TABLE 2. Proportion for Math Level

Section Size	Less than College Algebra	College Algebra or Higher	Total (n)
Small			
Freq	84	259	343
Proportion	.2449	.7551	
Expected	102	241	
Large			
Freq	219	455	674
Proportion	.3249	.6751	
Expected	201	473	
Total	303	714	1017
P_0	.2979	.7021	

H_0: $P_{lg} = .7021$ H_a: $P_{lg} \neq .7021$

Fail to reject H_0, since

$|Z_{calc} = -1.523| \rightarrow Z_{\alpha/2} = 1.645$

$$Z_{calc} = \frac{P_{lg} - P_0}{SQRT(P_0(1-P_0)/n)} = \frac{.6751 - .7021}{SQRT(.7021(.2979)/674)}$$

Table 3. Test Scores Statistics

	Test 1		Performance Score (average T2..T4)	
	Small Sect.	Large Sect.	Small Sect.	Large Sect.
n	343	676	343	676
Mean	101.66	100.96	99.22	93.20
Std. Dev.	15.59	15.08	14.50	15.69
t' score	-0.6862		-6.0923	
Deg. of freedom	676.5		737.4	
Prob > |t'|	0.4929		0.0001	
F	1.04		1.17	
Deg. of freedom	(342,675)		(675,342)	
Prob > F	0.6920		0.0969	

Table 4. Main Effects Analysis of Variance

Dependent Variable: PERFORMANCE R-Square = 0.564

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	13	141267.32	10866.72	99.95	0.0001
Error	1005	109259.87	108.72		
Total	1018	250527.19			
Model:					
Test 1	1	128176.97	128176.97	1179.00	0.0001
Sm Sec	1	7008.98	7008.98	64.47	0.0001
Math Level	3	2665.05	888.35	8.17	0.0001
Class	5	1122.21	224.44	2.06	0.0676
Instructor	3	2294.11	764.70	7.03	0.0001

TEACHING dBASE PROGRAMMING LANGUAGE IN INTRODUCTORY CIS COURSE: TRIALS, TRIBULATIONS, AND TRIUMPHS

Ali A. Nazemi, Roanoke College, Salem, VA 24153, (703) 375-2217

ABSTRACT

In recent years, because of advances and innovations in computer technology, IS educators have been confronted with the dilemma of selecting the appropriate topics for CIS curriculum. This decision has become even more profound in the introductory CIS/MIS course where the students must be exposed to a wide range of technologies. One topic, that has been the point of debate for many years, is the coverage of programming in this course. For many years, COBOL has been the corner stone of the CIS programming and it has not been until recently that other third generation languages such as BASIC, Pascal, and C have become more popular in CIS programs. The new technology and changing nature of business, especially the proliferation of the End-User Computing has prompted the movement towards more advanced and less procedure-oriented languages. Fourth Generation and Query languages seem to be the logical replacement for the current third generation languages [1,2]. In this paper, we will discuss the introduction of a dBASE language in the introductory MIS course and the tribulations and triumphs experiences in this undertaking. We will also discuss the advantages and disadvantages of using dBASE programming language both from teacher and student point of views. Finally, we will take a look at a comparison of third generation and fourth generation languages provide some hints for a smooth transition from BASIC to dBASE language.

INTRODUCTION

For many years, CIS instructors have been challenged with the decision of what programming language to use in the introductory CIS course. Because of the ever-changing business environment and evolving information technology, the coverage of topics in the introductory MIS course continues to evolve. The question is why do we attempt to teach programming in the introductory course? The rationales often cited covers a very wide range including:

- the need to make the students aware of what programmers go through in creating programs to
- improving students understanding of computers and software to
- teaching students logical thinking [3] to
- helping students in solving problems, and finally
- encouraging structured approach to writing, not only programs but any other type of writing [4].

However, the fact remains that in today's end user-oriented computing environment, it is essential for future end-users to gain adequate knowledge of programming in order to be a productive member of the information society. We, as educators, are responsible to ensure our students success by equipping them with adequate tools to enable them to perform more effectively and efficiently in the work place. The approach, discussed in this paper, addresses this very point and provides a set of guidelines for using dBASE programming language in the introductory MIS course.

The use of third generation languages has long been a common place in CIS curriculums. The selection of a specific programming language has been a direct function of the need expressed by the outside environment (particularly the companies hiring our graduates). This need, in turn, is often dictated by the type and the size of computers used by companies. Prior to the introduction of microcomputers, the major programming language for many EDP departments was COBOL. However, in the early 80's, the changes in technology has prompted a movement toward more user-friendly and microcomputer-oriented languages such as Pascal, C, and BASIC. BASIC, because of its availability on smaller system and ease of use has become one of the most popular languages for developing information systems on small computers. The CIS faculty eventually reacted by including BASIC in place of or in addition to COBOL in the introductory MIS course. A 1987 survey of 196 firms shows that for larger systems, the language of choice is COBOL while the smaller systems use BASIC and C as their primary languages [5]. However, in the case of companies with small computers, over 50% responded that they use languages other than the Assembly or third generation languages such as COBOL, FORTRAN, BASIC, C, Pascal, Etc... As the software technology evolves so do the needs of the business environment and as educators, we must be prepared to re-tool to satisfy these needs.

The movement towards a more end user-oriented languages is well under way in the business environment and the shift in academic institutions to teach such languages should shortly follow. This shift is also fueled by the direction the CIS programs are taking. This direction moves away from Computer Science-orientation to Business Administration-orientation. It is important to take into consideration the changes in end user computing environment which is towards less procedural and application-oriented languages [6]. These languages include such programming environments as Borland's dBASE III/IV, Microrim's R:Base, and Oracle's Oracle.

TRIALS

This paper describes the trials, tribulations, and triumphs of using dBASE language to teach programming in introductory MIS course. Prior to last year, the primary language used in this course had been BASIC which was covered over a span of 3 to 4 weeks. In fall of 1991, inspired by an article by Sasa Dekleva appearing in the Journal of Information System Education, the attempt was made to teaching programming portion of the MIS course using dBASE language. This change, however, was not done hastily or without taking into consideration numerous factors that had to be weighed against each other. As the first step, BASIC and dBASE were compared using several important factors. Each language was rated by faculty members teaching the course based on their familiarization and understanding of both languages. Exhibit 1 provides a summary of comparison of the two language environments based on a multitude of factors

The obvious advantages of BASIC are its availability, ease of use, and familiarity of most of the students with the language. It has the capability of handling arrays and in some new versions very good screen and graphics manipulation features. BASIC, however, lacks structure and modularity essential to a systematic structured programming process. It is quite cumbersome in handling sequential and random access files and has virtually no query capability. dBASE, On the other hand, is a non-procedural language that supports structured programming and application development. It supports modular programming through the use of procedure files and has a good debugging facility that provides a step by step progression through the program. One of the main advantages of dBASE over BASIC is that it can be written in ASCII/text format in practically any environment. However, one drawback is that dBASE code must be executed from within the dBASE shell. Another important factor to consider is how easily the sequential files can be generated and used in dBASE. The language itself is not that much different from BASIC especially with respect to the basic Input and Output statements. Even some of the control statements such as IF and WHILE work similarly in both environments and the Immediate Mode of BASIC is quite analogous to the Dot Command mode of dBASE. The variable assignment is also quite comparable using " = " as the assignment operator while dBASE also supports the STORE < Value > TO < Variable > statement used for the same purpose. One of the disadvantages of dBASE is the fact that it is limited to 256 memory variables and has no array handling capability (dBASE IV offers limited array handling). This, however, was not a factor at the introductory level of programming.

TRIBULATIONS

The transition to dBASE was not smooth by any means. The first major factor was familiarizing both students and teachers to dBASE environment. Since, the coverage of dBASE language is virtually non-existent in any of the leading MIS/CIS texts, a comprehensive set of notes and supplements had to be generated and made available to the students. The supplements had to be continuously revised to address pertinent material. Another early stumbling block was when students tried to test their programs. Since the dBASE text editor is in a separate environment than the Command (execution) environment, the repeated saves after each program modification was annoying at first but after a few programs it became much easier.

Another difficult factor encountered in conveying the programming process to students was the concept of dBASE's memory variables. Since all dBASE variables are similar as far as naming convention is concerned, some students had difficulty understanding and distinguishing among various variable types. This was especially true in cases where information had to be entered from the keyboard. In BASIC, the INPUT statement is used for both numeric and string variable types while in dBASE the Character variables must be received from keyboard using ACCEPT statement and the INPUT is used exclusively for Numeric variables [7]. To rectify this problem, the attempt was made early in the process to introduce @SAY/GET/READ command sequence for input and output statements. But the use of READ statement confused some students and the READ frequently interfered with some of the Format files of dBASE. The SAY command is used to position user prompts and values on a specific location on the screen and GET and READ are used together to capture user response from the keyboard.

In order to display information on the screen, BASIC offers PRINT statement and many of its

enhancements (i.e. PRINT USING, PRINT TAB, PRINT, , and PRINT;). In dBASE, there are several statements that can be used for this purpose each offering different characteristics and capabilities. These include, LIST, DISPLAY, SAY, and "?" statements. The one most resembling the BASIC's PRINT is the "?" statement which can display string literal and both numeric and character variables on the same line without any modifications. One of the drawbacks of this "?" statement, however, is its inability to display formatted output. To overcome this shortcoming, we had to introduce the use of some of the pre-defined formatting functions of dBASE such as INT(), ROUND() and STR(). Also @SAY command with PICTURE clause were used to show how this statement can be used to display formatted information in specific positions on the screen. The users must be careful when using @SAY statement for two reasons; one is that in order to display more that one memory variable or data field on the same line, all variables have to be converted to string, and the other problem is using @SAY in a loop. Since @SAY statement displays information on the same spot on the screen if the row and column numbers are not changed, it will overwrite the previous values on the screen if the same exact statement is repeated in the loop. To avoid this problem, we must either use a counter variable in place of row number that can be incremented, or use the row pointer function ROW()+1 in the place of row number to move the display one line down on the screen. When these variations in @SAY command was introduced, the complexity of programs drastically increased. This led us to exclude the @SAY statement from the list of commands covered in the following semester.

The control statements in dbase are very similar to those in BASIC. The dBASE's DO WHILE/ENDDO command is equivalent to BASIC's WHILE/WEND and works in similar fashion. In dBASE, however, we have the option of repeating the loop using LOOP statement or aborting the loop using EXIT statement. The IF-THEN-ELSE statement in BASIC is a bit difficult to understand especially when multiple statements must be used after THEN or ELSE. dBASE's IF/ELSE/ENDIF works similarly and has the advantage of having blocks of statements for Then and Else portions of the statement. The absence of THEN in dBASE was a bit strange at first and required some getting use to. Binary conditions in both languages are constructed in similar fashion using relational operators and conditions can be connected to other conditions using logical operators. The Logical operators in dBASE have to be accompanied by a leading and a trailing period(.) which caused some syntax errors in early programs. In dBASE, the DO CASE/CASE/ENDCASE set of statements came handy for multiple response questions and in constructing user menus.

Another problem faced was creating hard copies of program listing and the output. Statements such as SET DEVICE TO PRINT and SET PRINT ON can be very helpful, however, if the program is being run on a network server, there may be potential problems with piping the output to the right printer port. We had a little difficulty sending the output to the laser printer after we were already inside the dBASE. PCOL() and PROW() function can be used in place of COL() and ROW() functions (similar to LPRINT) to position output information on the printed copy but we never utilized these functions [7].

Perhaps the most important lesson learned from the first time use of the dBASE language was that introduction of too many Commands and Function added to the difficulty of learning to program. When a large set of commands were introduced in the earlier course, the students often became confused about the differences in statements and used arguments specific to one statement in other statements. For example, in occasions, some students tried to use row and column numbers with INPUT and ACCEPT statements or had difficulty with use of ? statement as opposed to DISPLAY or LIST. We overcame these difficulties by introducing a smaller set of statements that were sufficient enough for students to write fairly complex programs. Exhibit 3 illustrates a list of dBASE commands and their BASIC equivalents.

Another small but significant problem with using dBASE language was that some students did not have access to the dBASE package outside the lab or on their own personal computer. This caused many of them to put off working on their programming projects until they came to campus or until the lab was available.

TRIUMPHS

One of the most positive aspect of using dBASE language was how well it was received by the students. Even those who had a lot of difficulty with the programming logic and the use of statement were quite enthusiastic about the use of dBASE and were impressed with the capabilities of the language. Those with BASIC background made the transitions smoothly and were amazed at the similarity of statements in the two languages. Some of these students even expressed a sense of relief for finally getting rid of those pesky BASIC line numbers. For those without any programming background (and therefore no grounds for comparison), it was a bit difficult to get used to the environment but the learning curve improved drastically as they wrote more programs.

By introducing a very small subset of dBASE commands in the subsequent semester, we were able to

focus our efforts on the programming and logic design rather than on a broad-base coverage of dBASE statements. There were only 6 set of statements that the students were obligated to use:

1. INPUT .. TO
2. ACCEPT .. TO
3. DO WHILE/ENDDO
4. IF/ELSE/ENDIF
5. DO CASE/CASE/ENDCASE
6. ?

All programs could be easily constructed using this small set of statements. Other supplemental statements such as SET statements (i.e. SET DEVICE, SET TALK, SET STATUS, ...) and statements such as CLEAR, TEXT/ENDTEXT, WAIT, SKIP, LOOP, EXIT, and RETURN were introduced as they were needed.

Another advantage of using dBASE language, was the absence of GOTO statement that since has been one of the major disadvantages of BASIC as far as structured programming is concern. The GOTO-less programming environment forced the students to plan ahead for their loop and iteration operations hence following a top-down structured approach to designing programs. The use of CASE statement was also very successful as it drastically reduced the need for using Nested IF statements. Some students used the OTHERWISE option of the DO CASE statement to communicate back with the user and provide an interactive environment.

The programming portion of the MIS course, by design, preceded the discussion of Data Base Management Systems and dBASE III/dBASE IV's data manipulation and management capabilities. We introduced the students to the use of data files by designing the assignments so that existing programs, with very small modification, could read the information directly from data files. This reduced the time for entering data from keyboard and helped in explaining the concept of Batch versus Interactive data processing. In order to make it easier to understand, we simply matched the memory variables received from the keyboard with field names in the data file and eliminated the ACCEPT and INPUT statements. Exhibit 1 and 2 illustrate the two versions of the same program; one using interactive ACCEPT and INPUT commands and the other using dBASE data files. In the subsequent weeks, the students learned how to create and populate database files and prior knowledge of the programming made the coverage of Query statements much easier.

By far the most triumphant point of the course was when the students unanimously agreed that they would more likely end up using dBASE or similar software to develop programs than BASIC outside the school and appreciated the fact that we took time to incorporate dBASE programming into CIS course. This was very important is helping to keep students enthusiastic about the programming. In fact, some working students have already started using the knowledge gained in the classroom in their place of work to create specialized reports for the existing dBASE files.

EXHIBIT 1
Interactive Payroll Program

```
**************************************************
* PROGRAM  : PAYROLL1.PRG  - Payroll calculations
* AUTHOR   :                With Control Stmts.
* DATE     : Spring 92
**************************************************
SET BELL OFF
SET TALK OFF
SET ECHO OFF
SET STATUS OFF
More = 'Y'
DO WHILE UPPER(More) = 'Y'
  CLEAR
  ********* Interactive Input Statements
  ACCEPT "Enter Employee Name: " TO Name
  INPUT "Enter Hours Worked: " TO Hours
  INPUT "Enter Rate of Pay : " TO Rate

  ********* Constant Assignment
  TaxRate = 0.23
  DedRate = 0.1

  ********* Control statement to check hours
  IF Hours>40
    Gross = 40*Rate + (Hours-40)*1.5*Rate
  ELSE
    Gross = Hours*Rate
  ENDIF

  ********* Processing Calculations
  Tax   = Gross * TaxRate
  Deduct= Gross * DedRate
  Net   = Gross - (Tax + Deduct)

  ********* Output Statements
  ? "  Gross Pay     Taxes     Deductions   Net Pay"
  ?
  ? Gross, Tax, Deduct, Net

  ********* Loop control statement
  ACCEPT "More Employee Data (Y/N)?" TO More
ENDDO
```

EXHIBIT 2
Batch Payroll Program

```
*************************************************
* PROGRAM  : PAYROLL2.PRG  - Payroll calculations
* AUTHOR   :                using PAY.DBF
* DATE     : Spring 92
*************************************************
SET BELL OFF
SET TALK OFF
SET ECHO OFF
SET STATUS OFF
TaxRate = 0.2
DedRate = 0.1

CLEAR
********* Column Headings
?"NAME              ","GROSS   PAY","   TAXES
","DEDUCTS.","NET PAY"

********* Open PAY file & read record until End Of File
USE PAY
DO WHILE .NOT. EOF()

 ********* Control statement to check hours
 IF Hours>40
   Gross = 40*Rate + (Hours-40)*1.5*Rate
 ELSE
   Gross = Hours*Rate
 ENDIF

 ********* Process Calculations
 Tax   = Gross * TaxRate
 Deduct= Gross * DedRate
 Net   = Gross - (Tax + Deduct)

 ****** Output statements using STR function to format
 ? Name, STR(Gross,7,2), STR(Tax,7,2),
   STR(Deduct,7,2), STR(Net,7,2)

 ********* Pointing to the next record
 SKIP
ENDDO
```

EXHIBIT 3
BASIC and dBASE Statements

BASIC Statement	dBASE Equivalent
INPUT	INPUT .. TO ACCEPT .. TO
PRINT	?
IF-THEN-ELSE	IF/ELSE/ENDIF
WHILE/WEND	DO WHILE/ENDDO
READ/DATA	USE data-file name

EXHIBIT 4
Summary of Comparison of BASIC and dBASE

FACTOR	DBASE	BASIC
Availability	High	High
Ease of Use	Moderate	Low
Support for Structure	High	Moderate
Modularity	High	Low
Speed of Execution	Low	Low
Program Editing	High	High
Graphics Capabilities	Low	Moderate
Data File Handling	High	Low
Debugging Capabilities	High	Low

REFERENCES

[1] Augustine, Jr., Fred k. and Surynt, Theodore J.,"A Fourth Generation Approach to the Introductory Programming Course", *CIS Educator Forum*, Volume 2, Number 2, Winter 1990.

[2] Whitt, Jerry d., "4GL and DBMS coverage in the introductory MIS Course: A Survey", *Interface*, Volume 10, Issue 1, Spring 1988.

[3] Saret, Laura E., "The Programming Logic Course in the Data Processing Curriculum", *Interface*, Volume 11, Issue 2, Summer 1989.

[4] Dekleva, Sasa M., "Introduction to Programming Using dBASE", *Journal of Information Systems Education*, Volume 3, Number 2, Fall 1991.

[5] Hatch, R.A., Koster, A., and Marder, S., "Selection of Programming Languages for the Computer Information Systems Curriculum", *Interface*, Volume 10, Issue 2, Summer 1988.

[6] Te'eni, Dov, "Using Lotus 1-2-3 and dBASE III to Teach Programming in an Introductory MIS Course", *Interface*, Volume 11, Issue 1, Spring 1989.

[7] Wray, Robert A., *dBASE III Plus Programming*, Second Edition, Boyd and Fraser Publishing, 1991.

A FIRST COMPUTING COURSE PROJECT: TURNING STUDENTS INTO POLLSTERS

BY

Stuart A. Varden
School of Computer Science an Information Systems
Pace University

ABSTRACT

This paper describes a class project that integrates several key themes and skills of a typical first computing course for undergraduate non-computer majors. Students design, conduct, analyze and summarize a campus student opinion survey. The project illustrates the many contributions that computing can make at nearly every stage of a complex undertaking.

INTRODUCTION

Have you ever been faced with a class full of undergraduate non-computer majors who are taking that one required computer course that stands between them and graduation? At Pace University, that course is called "CIS101 - Introduction to Computing" and the author has struggled with it on many occasions. CIS101 is a service course to the schools of business, arts and sciences, and nursing.

An ongoing challenge in such a course is to maintain student interest while covering the required material. Many successful approaches to promote student interest have been reported in the literature, such as the use of multimedia, going on field trips, using case studies and role playing, and even dividing the class into two teams to compete in a game of "computer trivial pursuit." Most of these approaches, however, either focus mainly on the computer itself or have a certain artificial feel to them.

The author, instead, felt it was important to find a practical application of computer technology that would involve a blending of specific computer competencies with a clear illustration of how computer technology can become a powerful tool to help both individuals and organizations accomplish useful work. In particular, the author was looking for an activity that would:

* Integrate several major themes and skills of the course through the practical application of computer technology to solve a real problem.
* Involve the active participation of all students.
* Be of interest to students.
* Result in a tangible product or outcome.
* Be accomplished during the semester.
* Cost little or no money.

THE STUDENT OPINION SURVEY PROJECT

After several failed attempts, the author finally hit on the idea of conducting a student opinion survey using the students to gather, process and analyze the data. The

fact that a Presidential election year was approaching seemed to make the idea even more appealing. If the project was successful and revealed some interesting results, it might make a good article for the campus newspaper.

The project seemed to meet all the criteria that the author had identified: there would be plenty of opportunities to relate the project to the course content; it should be of interest to students to discover how their fellow students feel about issues; the project would produce a tangible result; it probably could be completed in a semester; and finally, it should cost next to nothing.

GETTING THE PROJECT UNDERWAY

In Fall, 1991, the author had been assigned to teach two sections of CIS101. Although it was anticipated that there would be some logistical difficulties in having both sections work on the same survey, the author felt that by combining the labor pool of the two sections a larger survey sample size could be achieved. A total of thirty students would be available to conduct the survey.

The students in both sections were presented with the idea of the project during the fourth week of the semester. The author reviewed with the students a checklist of tasks that would have to be performed. Students were asked to "sign up" for selected tasks, while other tasks would involve the participation of all students. The checklist included some work for the author as well.

The first two weeks were devoted to deciding what issues the survey should include. After some discussion, the students arrived at the idea of surveying students on five broad societal issues and five local campus issues. The students selected the issues, and the author helped the students phrase them in a form that would allow them to be responded to on a five-point scale running from "Strongly Disagree" to "Strongly Agree" (see Table I). The author also recommended that a few demographic facts be included, such as sex of the respondent, so that it would be possible to compare response patterns between different groups within the sample.

Students were told that the survey would not be "scientific" since the sample would not be random. Nevertheless, hopes were high that the findings would be both interesting and reasonably representative of the total student body of the campus. The next two weeks were spent conducting the surveys.

INSTRUCTIONS TO STUDENT POLLSTERS

All students in the two classes were given a training session on how to administer the survey. This provided a good chance to emphasize that the findings of the survey can be no better than the qualify of the data that go to make it up. The training session also was designed to promote inter-surveyor reliability.

PROJECT LEARNING OPPORTUNITIES

A surprising number of learning opportunities that relate directly to the course content emerged from the Student Opinion Survey project, including several

TABLE I: SURVEY OF STUDENT OPINIONS

SOCIETAL ISSUES

1. The sale of narcotics should be made legal in the U.S.....: SD D U A SA
2. Capital punishment should be made legal in New York State.....: SD D U A SA
3. Abortion should be made illegal, except when the woman's life is in danger.....: SD D U A SA
4. Smoking should be outlawed in all public places.....: SD D U A SA
5. The killing of animals for their fur should outlawed.....: SD D U A SA

Pace CAMPUS ISSUES

1. The suggested proposal to move all business courses to the Pace White Plains Campus should be approved.....: SD D U A SA
2. Pace University should devote more emphasis to improving the competitiveness of its athletic teams in order to gain a national reputation for sports.....: SD D U A SA
3. Course text books should be changed less frequently so that students can buy cheaper used texts, even if it means that the texts are a year or two out-of-date..: SD D U A SA
4. I would be willing to pay a little more for better quality food service at the cafeteria.....: SD D U A SA
5. First year students living in the dorms should not be allowed to keep a car on campus in order to promote more on-campus social activities.....: SD D U A SA

DEMOGRAPHIC DATA

Sex.....: Female ___, Male ___

Enrollment: Full Time ___, Part Time ___

Residence.: Live on Campus ___, Commuter ___

that the author had not anticipated. The main ones are reviewed below.

Project Management

The students were presented with a list of tasks that

would have to be completed to accomplish the project. This provided an excellent vehicle for discussing the need to plan, staff, organize and control a complex project. After examining the tasks, it became clear to the students that certain tasks could be conducted in parallel, while others must wait for earlier tasks to be completed. This experience helped set the stage for discussing systems development methodologies covered later in the semester.

Wordprocessing

Both the survey instrument and newspaper article were prepared using wordprocessing software. As repeated editing, reformatting, and spell checking was required, the advantages of using a wordprocessor quickly became clear.

Record Layout and Coding

Once the class had decided on the issues and demographic factors to be included in the survey (see Table II), it was necessary to design the format and coding scheme for the data before data entry could begin. This need provided a practical context for learning about the structure of fixed-length records and flat files.

Data Gathering and Entry

Each student received ten copies of the survey instrument and was given two weeks to return the completed surveys. A total of 300 returned would have been expected if all thirty students had returned all surveys. As it turned out, 195 completed surveys were returned.

Three students had signed up for the data entry task. This required them to learn the basics of the XEDIT editor available under VM/CMS. Once the data had been entered, the three students had to learn how to send the files to a common user ID where the data would be processed. Once there, the three files had to be joined into a single file. This provided a further opportunity to discuss files and how they can be managed.

E-Mail

Since the students working on the project were divided between two CIS101 sections, it might have been difficult for students to collaborate with one another. Many students were commuters, and very few students in one section knew anyone in the other section. Yet once they were introduced to the idea of electronic mail and had been issued user codes on the University's mainframe, students who needed to coordinate their activities simply left E-mail messages for one another. In fact, the two students who wrote the article that appeared in the campus newspaper never actually met each other!

Statistical Processing of Data

The data were processed using the Statistical Package for the Social Sciences (SPSS). This allowed a comparison of the advantages and disadvantages of using packaged programs instead of writing programs in a third-generation language. Two students were responsible for entering the required SPSS command statements and generating the statistical reports.

Data Analysis and Interpretation

All students received a copy of the SPSS report. Once the format of the report and certain statistical concepts were explained, the students were asked to interpret the meaning of the survey findings. This lead to a discussion of how pollsters, market researchers, and advertising firms use statistics to study voter and consumer trends. From here it was natural to introduce the role of an MIS and the idea of information as processed data.

Micro-to-Mainframe Link

The output created by SPSS on the mainframe is in the form of an ASCII file, while the article summarizing the results was being prepared using PFS:Professional Write on a microcomputer. Since the students had learned about up-and-down loading of files in class, they realized that it would make sense to down load the desired statistical tables from the SPSS ASCII file. This meant that the time consuming and error prone task of keying in the tables could be avoided.

Document Conversion

After the article was completed, it was learned that the campus newspaper, New Morning, delivers its text copy to the printer in the form of a WordPerfect 5.1 file. This presented no problem, since Professional Write can save a file in WordPerfect format. This led to a discussion of some computer industry issues, such as standards and the competitive forces at play in the highly volatile PC-software marketplace.

OUTCOME OF THE PROJECT

At first, many students did not seem to understand why they were doing a student opinion survey as part of a computer course. Only as the project unfolded over the semester did the connections become clear. The author pointed out to the students that the survey could have been conducted without any aid of a computer whatsoever. Nevertheless, the students could see that computer technology has a useful role to play at nearly every stage of the project.

The one disappointment was that the project got a little behind schedule. The author had hoped to have the article in the hands of the students by the end of the semester. Instead, the article summarizing the survey did not appear in the campus newspaper until early in the following semester.

The survey findings proved to be quite interesting. There were a few surprises that have catch the attention of several parties from within the University, including the administration. But that's the topic of another paper.

INTERNAL MARKETING

An interesting sidebar to the curriculum issues surrounding the first (and probably only) computing course that non-computer majors take is its potential as a recruiting device for computing majors and minors. At Pace University, students usually take CIS101 in their freshman or sophomore year, and many of them have yet to settle on a major. When students take CIS101, it is the only time that large numbers of students and the Information Systems

department come face to face. Although there is no active campaign by the department to proselytize students to become computer majors or minors, it is true that a dull class is not likely to sway the undecided student in the direction of computing.

On the other hand, most students are pleased to be identified with an interesting and visible project such as the Student Opinion Survey. The appearance of the survey results in article form entitled "CIS Poll of Student Body" on the front page of the campus newspaper says to the whole University community that people who take computer courses do interesting and useful things that produce results.

It is difficult to measure the influence that the project may have had on the future course selection decisions of the students in the two CIS101 sections who conducted the survey. The author, however, can relate one brief comment offered by a former student who was hurrying to class:

"I was really excited when I saw the survey results in the campus newspaper. It was great to see our work finally in black and white. But you will have to excuse me. I'm late for my programming class."

The student had been an accounting major the previous semester.

COBOL's Future: Better Wear Your Sunglasses!

by

Mark W. Smith, Ed.D.
Assistant Professor
&
Michael J. Payne, M.B.A.
Assistant Professor

Purdue University
Department of Computer Technology
Knoy Hall, Room 242
West Lafayette, IN 47907
FAX: (317) 496-1212

BITNET address: msmith@purccvm
INTERNET address: msmith@vm.cc.purdue.edu
Phone: (317) 494-5125

BITNET address: payne@purccvm
INTERNET address: payne@vm.cc.purdue.edu
Phone: (317) 494-2566

ABSTRACT

Is COBOL dying? By no means! The future of this thirty year old language is very bright! It is changing with the times. At first, COBOL was developed to provide a machine independent language for application software development. Today COBOL is everywhere! It's even on PCs and UNIX. It supports additional intrinsic functions, works under WINDOWS and allows GUI. As to the future, COBOL++ (Object Oriented COBOL) is now available in an experimental form. This paper will discuss how COBOL is changing with the times, and how it continues to be a major competitor in the application software development marketplace.

INTRODUCTION

That's right, the future looks extremely bright for COBOL. To borrow from a famous saying "reports of COBOL's demise have been greatly exaggerated." Rather than the aging dinosaur depicted by its detractors, we prefer to think of COBOL as Peter Coffee says in comparing COBOL to the shark, ". . . an ancient but highly effective design, honed by constant competition into a variety of forms, each of which has some special strength." (4)

The recent interest in languages like C, Pascal, and BASIC, which all offer varying degrees of sophistication, and other languages utilizing concepts of Object Oriented Programming (OOP) like C++, certainly make them popular. However, COBOL has grown and evolved, incorporating many of these features, including structured/modular programming as well as other enhancements. This ability

to adjust and evolve make COBOL's future look very bright indeed. (1, 14)

COBOL BACK THEN

This paper attempts to present a brief background on COBOL, its evolutionary changes, and a look at where COBOL is going in the future. Remember that COBOL was developed as one of the first machine independent computer languages for application software development. Much like FORTRAN was developed for science and engineering applications, COBOL was tailored for business applications.

We have seen COBOL evolve through several important phases resulting in standard COBOL syntax and compiler versions. Of course, COBOL 68 was a major step in that programmers could learn COBOL and depend on the fact that if they went to a different project or different company they wouldn't have to relearn their skill. COBOL 68 had some problems that led to a new version, COBOL 74. The vast majority of business programs were developed using COBOL 68 and 74. Estimates of the billions of lines of COBOL code in existence have been made. (7) However, this is not the reason that COBOL continued in popularity, although it was probably a contributing factor.

There were other reasons that contributed to COBOL's increased use such as:

* COBOL 74 had many benefits over COBOL 68

* COBOL became the de facto language for business programming since it was developed for accounting (24)

* a large group of programmers were already familiar and trained in COBOL

* COBOL is the single most popular commercial programming language in existence (8)

COBOL NOW

With the increasing demands for structured style and the need for 3 GL's to compete against 4 GL's, COBOL undertook another evolutionary change. COBOL 85 standards were adopted, and once again COBOL remained competitive and adaptive. SQL, DB2, and continued CICS support by COBOL insured its resurgence as the main software development language of the late 80's.

Not only does COBOL 85 provide general upward compatibility of earlier versions, but "new features included in COBOL-85 are equivalent to those that have proven to be extremely popular in other programming languages such as BASIC, Pascal, Modula2, and C." (12) In other words, COBOL 85 brought into COBOL program development much of the flexibility its critics argued was missing. True structured commands were introduced; in-line performs, the case structure, and many scope terminators.

Of course, a more recent trend has been the adaptability of COBOL 85 to the microcomputer environment. With the advent of "right-sizing" COBOL has once again proven itself a worthy development language. Not only can you develop and test your COBOL applications on a microcomputer, but you can then upload and put them almost immediately into production due to COBOL standardization across platforms. (17) This new trend allows for multiple file structures, full testing and debugging with an outstanding set of tools including comprehensive interactive source debugging, support for screen building, CICS and SQL support and even the generation of

structure charts. (22) But what about the really new trends starting to make the news such as Object Oriented Programming (OOP), Graphical User Interface (GUI), and continued portability between platforms and operating systems?

Some of these new PC COBOLs and IBM 370 COBOL support intrinsic functions found in many other languages. These intrinsic functions include: business functions, mathematical/statistical functions, trigonometric functions, and even character/string functions! (8,9)

COBOL's adaptability to the GUI world of OS/2 Presentation Manager and Windows is now available according to both the Microsoft and Micro Focus companies. (6,19,20) Not only is COBOL able to be run from Microsoft Windows, but you can develop programs to work and run in the Window's and PM environment. (6,15,20)

COBOL OF THE FUTURE

The evolution continues! In November 1989, the CODASYL Committee formed the Object Oriented COBOL Task Group (OOCTG) to begin the development of new constructs for object oriented COBOL. (8,10,11) One of their major challenges is to produce a new standard in COBOL that will allow OOP techniques as well as compatibility with older versions. Lance Eliot, who heads an information technology consulting group in Long Beach, California, believes that the future of COBOL through the 1990's is "readily assured" with this new version. (7) This new version of COOBOL (COBOL Object-Oriented Business Language), COOL (COBOL Object Oriented Language), or OOCOBOL (Object Oriented COBOL) will have the features of object classes, methods, inheritance, etc. (11) Some benefits which many hope will be realized include:

* Rapid prototyping and reusability
* Easier maintenance
* Packaged objects
* Faster systems development

Ultimately, it is hoped this will lead to more reliable application software. Of interest, the Micro Focus company now offers an experimental version objected oriented version of COBOL. (18,21)

Other future COBOL changes include the use of preprocessors that allow you to write COBOL statements to access a relational database without using SQL. (2,3). It reads the COBOL statements accessing the database and generates the COBOL code into the appropriate SQL statements needed in order to access the database.

Of course, many have said that 3 GL's will disappear due to 4 GL's. Ketler and Moncada make a good argument for the obsolescence of 3 GL's and in particular COBOL, based on IBM's new Systems Application Architecture (SAA), and the focusing of IS functions more on the PC environment. (13) Although this may happen, we believe it won't. SAA fully supports the use of COBOL and for speed, a compiled COBOL program compared to a 4 GL application dealing with traditional report based requirements, is hard to beat. (5,19,23) Along with this, is the fact that COBOL is fully integrated into the PC environment with the advent of Micro Focus Workbench COBOL and other excellent PC based COBOL compilers and tools.

The fact is, COBOL just hasn't gone away! Of interest is that many code generators and CASE tools generate COBOL code. (16) This presumes that COBOL is going to be around for yet a little while longer.

CONCLUSION

Although COBOL certainly isn't the easiest language to use, it doesn't look as if it's going to disappear anytime soon. Often seen as the dinosaur that won't die, COBOL continues to evolve into an extremely sophisticated application development language. New tools, compatibility with multiple platforms, CASE compatibility, OOP techniques, and GUI interfacing all point toward a renewed use of COBOL. Now that it has again adapted and evolved to the demands of software developers COBOL appears to have a very bright future indeed!

REFERENCES

1. Appleby D., "COBOL (continued use of Common Business Oriented Language programming)," Byte, Oct 1991, 16(10), pp. 129+.

2. "Acucobol introduces Acu4gl to open up SQL databases to COBOL," Computergram International, Nov. 6, 1991, n1799.

3. "Accucobol translator links COBOL programs to 4GLs," Grygo Gene, Digital Review, Dec 2, 1991, 8(36), pp. 26.

4. Coffee, P., "COBOL evolves as mission-critical tool; six DOS-based compilers answer growing need for multiplatform application," PC Week, January 21, 1991, 8(3), pp. 75+.

5. "COBOL/2 becomes IBM's SAA workstation standard," Computergram International, May 9, 1991.

6. "Dialog System 2.0 Streamlines User Interface Development," COMPILATIONS, July/August 1991, p1+.

7. Eliot, L., "Do object oriented techniques and COBOL mix?", DataBase Programming and Design, January, 1992, pp. 56-60.

8. Garfunkel, J., "COBOL in the 90's: Mature, Modern and Much Maligned," Computer Standards & Interfaces, June 1991, 10(6), pp. 17+.

9. Garfunkel, J., "COBOL Intrinsic Functions," Micro Focus, December, 1990.

10. Garfunkel, J., Phone Calls on March 5 & 6, 1992, Member of CODASYL, ICO and OOCTG.

11. Garfunkel, J., "Object Oriented COBOL", Micro Focus, May, 1990.

12. Headrick, W., and Roberts R., "COBOL-85 and structured programming: opportunities to improve programming style", Proceedings of the Eight Information Systems Education Conference, October, 1990, pp. 28-31.

13. Ketler, K. and Moncada, S., "The role of third generation languages past, present, and future", Proceedings of the Eight Information Systems Education Conference, October, 1990, pp. 192-196.

14. Keyes J., "Backers of 'open COBOL' fight 'oxymoron label': standards groups continue to hammer out portability , but liberties over the years challenge open direction", Software Magazine, Nov 1991, 11(13), pp. 90+.

15. Kilcoyn, V., "Three COBOL jewels?", EXE, August 1991, v6 n3, p21(3).

16. "LAN-based CASE tools for DOS and OS/2", PC Week, April 8, 1991, 8(14), pp. 112.

17. McMullen, J., "Why PC COBOL is gaining ground"; Datamation, May 15, 1991, 37(10) , p70-72

18. "Micro Focus' forthcoming Object COBOL looks likely to keep IBM sweet in the UK company", Computergram International, April 18, 1991, n1655.

19. "Micro Focus Signs Strategic Agreement with IBM", COMPILATIONS, July/August 1991, pp. 1+.

20. "Microsoft gets COBOL into Windows", Data Base Advisor, Nove 1991, v9 b11, pp. 111.

21. "Object-Oriented option", Computer Language, Nov 1991, v8 n22, pp. 137.

22. Payne, M. and Smith, M., "COBOL on Microcomputers: A Quality Curriculum Option for Teaching Business Oriented Applications Programming", Proceedings of the Eight Information Systens Education Conference, October 1990, pp. 24-27.

23. Rohrbough, L., "New for PCs: COBOL for IBM's AD/Cycle", Newsbytes, Sept 16, 1991.

24. Winship, S., "Buyers consider 'Dinosaur' COBOL in tune with times", PC Week, January 21, 1991, 8(3), pp. 75+.

ICCP Examination Review Materials: An Evaluation

Eli Boyd Cohen, CDP, CSP CCP
Eastern New Mexico University

Thomas Callan, CDP, CSP
Caterpillar, Inc.

Abstract. In 1991, the Data Processing Management Association (DPMA) set up a committee to evaluate the available published ICCP examination review materials. The authors developed an evaluation procedure that was used by that committee. The purpose of this paper is to describe the evaluation procedure and to indicate the results of the committee's evaluation.

Using a form of illuminative evaluation, the committee evaluated the suitability of the only materials available at the time, Bird's tapes and printed figures and the text offered by SIG-CP, for use by individuals and in a group setting. The committee found differential areas of strengths and recommended situations in which each would be preferable and in which both should be used together.

Introduction

The Institute for Certification of Computing Professionals (ICCP) examines individuals seeking credentialing in programming, analysis, and/or data processing management. Various publishers provide materials designed to assist such individuals develop their understanding of the areas tested. The task of the publisher has been complicated by ICCP's continuing refinement of its tests. For the individual candidate, the task has been further complicated in that these published materials have not been evaluated. Anecdotes of wrong answers in a published test review book, for example, demonstrate the need for such an evaluation.

In 1991, the Data Processing Management Association (DPMA) set up such a committee to evaluate the available published ICCP examination review materials. Table 1 lists the committee members. The authors of this paper designed the evaluation procedure used by the committee. The purpose of this paper is to describe the evaluation procedure and to indicate the results of the committee's evaluation.

To understand how the evaluation process was conducted, we first turn our attention to the various alternative types of evaluation and to the types of Information Systems (IS) professionals who might be interested in these materials.

Types of Evaluation

Program evaluation came into vogue during the 1960's as the government demanded evaluations of major social and educational projects as a condition of the funding. These projects employed teams of evaluators and thus gave birth to the new field of program evaluation. Contributions to this field came from the fields of statistics, administration, and philosophy.

Most commonly, the goals of evaluation are classified as formative and/or summative in nature. Formative evaluation is used to form, develop, and improve that which is being evaluated. For example, a professor interested in determining what pages of a manuscript are in need of rewriting might conduct formative evaluation, asking reviewers to indicate where the manuscript is confusing.

In contrast, summative evaluation is used to determine to what extent the object of evaluation succeeded in reaching its goals. Professors giving finals to determine the students' grades use this form of evaluation.

The difference between these forms of evaluation lies not only in their purpose, but also in how they are conducted. Gathering information on the process of the object being evaluated and how that object can be improved constitute the core of formative evaluation, while summative evaluation focuses its attention on outcome data.

These two primary goals of evaluation have been expanded to include other evaluative techniques, techniques that seek to answer other questions. One such technique is Illuminative Evaluation (IE). IE's goal is to describe the process or program against the needs of the various users or customers of the program or process. The outcome of IE is not a simple statement that Program A is or is not worthy, nor it is merely statements on how Program A might develop to meet better its goals. The outcome of IE is a combination of these two goals: a description of the Program A, for whom it is appropriate, for whom it is not, and how it meets and fails to meets the its own goals.

Our evaluation of ICCP examination study materials uses Illuminative Evaluation. To do so, we must first understand who are the customers in this study.

The Customers

By definition, the customers of ICCP examination review materials include those who desire to sit for one or more of these exams. Only IS professionals who have worked five years as a programmer, analyst, or so on can sit for the examination. However, the customers can be further classified.

One obvious classification is by the examination for which the individual wishes to sit. ICCP exams include a core examination (material common to all exams) and separate specialty tests. The CDP, CSP, and CCP exams require the individual to sit for a specialty exam in Management, Systems Development, and Procedural Programming respectively, as well as one more specialty exam. Currently these other specialities include Business Information Systems, Communications, Office Information Systems, Scientific Programming, Software Engineering, and Systems Programming. Different customers would be sitting for different exams.

Another area in which customers differ is their background and preparation. Clearly, those individuals who are already relatively prepared for the exams need less thorough coverage of the reviewed material than those who are learning it for the first time.

A third differential of the customer is by learning style. Some people have a preference for (and learn better) by reading while others are more aural learners. Still others learn better by speaking and doing.

This leads to a related way of differentiating customers. Some groups, such as DPMA chapters, are the ones who purchase the materials to enable them to put on group sessions. Some ways of presenting review material lend themselves more easily to group presentation than others.

We can summarize these classifications in the following framework:

1. Topic,
2. Review/First Time,
3. Learning/Teaching Style, and
4. Group/Individual.

We will use this framework in our evaluation.

Procedure

The evaluation incorporated the following steps:

1. Development of a Rating Instrument
2. Rating of the Materials
3. Interpretation of the ratings.

These steps are discussed below.

Development of a Rating Instrument

The framework described above was incorporated into a rating scale as seen in Tables. 2 and 3.

Topic and Review/First Time. The *ICCP Official Study Guide* provides an outline of topics that will be included in each examination. The topical coverage was measured by determining to what extent the materials being evaluated covered the topic. Coverage of each topic was measured using a five point Likert scale.

To understand coverage you need to know that the *ICCP Official Study Guide* provides details on what is expected to be covered. While the rating sheet asks for an evaluation of coverage on items down to the second outline level (e.g. 1.1), the ICCP Guide details the items beyond the third outline level (e.g. 1.1.1).

To help ensure inter-rater reliability, each of the five alternatives on the Likert scaled were behaviorally anchored, as follows:

Unacceptable means provides no coverage.

Less Than Average means provides coverage only in passing, of little depth, whether or not current.

Average means provides coverage of some topics, but few in depth and is current.

Acceptable means provides at least 75% of the topics listed in the ICCP Study Guide and these are in depth and current.

Comprehensive means provides coverage of all constituent topics in depth and current.

This scaling is a compromise. Ideally, three separate scales should be used, to determine broadness of coverage, depth of coverage, and currency of coverage. This idea was discarded as unworkable within the real-world constraints under which we operate. The form used by each rater is shown as Table 2.

Learning/Teaching Style. The Learning/Teaching style supported is determined by the mode of presentation (audio tapes, textual material and so on) and so need not be evaluated by raters.

Applicability to Group and Individual Use. To rate the potential effectiveness of each course for group and for individual use, the rating scale contains items designed to measure suitability in each of these areas.

As seen in Table 3, each of these items is measured on a three point Likert scale: feature is not offered, offered (but not exceptional), and offered and well done. An example may help differentiate between rating two and three. Consider an index of topics. If it is offered, but is only, say, twenty entries long, it would receive the second rating. A two hundred entry long, cross-referenced index would be rated as three, well-done.

Recognition of Potential Bias. Each rater was also asked to recognize the potential for his or her bias, due to prior use of or other association with the material under review.

Rating of the Material

At the time of evaluation, only two sets of materials were on the market, Bird Certification Review (2320 Lynx Way, Boise, ID 83705 (208) 384-1600) and DPMA SIG-DP materials (Bob Radford, CDP, CCP, 751 Airport Rd, Monterey, CA 93940 (408) 375-8296).

The Bird materials consists of audio tapes and accompanying printed illustrations. The printed manual also contains extensive coverage of how to teach to a group using these materials. These materials are dated 1990. The costs are $295 for each course (CDP, CSP, CCP) or $450 for a group/chapter study package. In addition, computerized question and answer drilling is available for $129 for each certificate, and additional workbooks cost $35.

The SIG-DP materials consist of a text and overhead masters. The materials reviewed here are dated January, 1991 (version 90-3.00) and will be updated to include material not currently covered. The costs are $50 for each student manual and $35 for the instructor's packet of overheads.

It is expected that the ratings shown here will be out-of-date as soon as one of these authors revises the materials. (It is our hope that our evaluation will be used by authors of ICCP examination review materials to develop and improve their products.)

Members of the committee, listed as Table 1, served as raters. Each member rated each course in terms of usefulness for individual and for group instruction. In addition, at least two committee members rated each topical area. The raters of specialty areas are certified in those areas.

The Results

The matrix shown as Table 4 summarizes the responses of the raters. Table 5 shows the summarized ratings on group and individual study parameters.

The number of raters is too small to apply the statistical techniques for determining inter-rater reliability and scale reliability. However, these techniques are not needed because of the goal of this study. We will not be asserting that one set of materials is superior to the other. Rather, we intend to interpret these results to indicate the strengths and weaknesses of each and to assert when one might be preferred to the other.

Interpretation of Results

SIG-CP. The raters agreed that the materials in the SIG-CP packet in the version presented here were far from complete and so were inadequate as a stand-alone course. For example, the section on "Associated Disciplines" (accounting, statistics, and so on), which typically had the lowest pass rate on the examination in its earlier forms and so is the most crucial for review, was not included in the version reviewed here. The manual informs us that these inadequacies will be remedied in the next release.

However the raters did find that having a text with written, easily reviewable content is of great value. Consequently, the reviewers found substantial value to this approach.

The overhead masters provided as the instructor's "manual" should improve in later editions to the point where they provide explanatory value. At the moment, the overheads serve primarily as word outlines of what is being covered. We recognize that every journey begins with a first step.

Bird. The Bird materials have a professional look-and-feel. The tapes with figures are useful for individuals who wish to review material they already know. The use of tapes is especially useful for those who spend a lot of time commuting and for those who learn best aurally. However, in the car, one cannot easily review the figures.

Bird's tapes and figures can also be used in group sessions, and are particularly helpful if the instructor is less experienced with teaching this material.

The raters feel that these materials could be made even better through the following ways:

1. Currently there is no easy way to skim the material. Providing an extensive index of topics with the appropriate tape index number could enhance this. (We recognize that different tape players show different numbers. However, these numbers are more-or-less proportional to one another. After establishing the appropriate multiplier, an individual can tailor the tape counter number shown in the index to his or her own tape player.)

2. The figures that are in the materials are good. However many major points are not covered by figures. We would like to see an even closer correspondence between the tapes and the figures.

3. For learning new material, our reviewers felt that the written word was a more appropriate medium. The package is better suited for reviewing information that one had once learned.

Conclusions

It was the consensus of our reviews that, once the SIG-CP materials are complete, the safest route for studying for the ICCP exams will be to use both Bird and SIG-CP materials, as they complement each other.

The Bird materials are best in the case where an individual who has prior education in the material needs a full, more current review. The written SIG-DP material are best for learning the material for the first time. Their indexing provides for quickly checking on specific topics, as well.

Currently, if a group instructor needed to choose just one course, the Bird course is better for the instructor who is not familiar with teaching the subject. Bird's instructions to the instructor on how to set a class are excellent; Bird also deals with adult education and its challenges.

In contrast, an instructor who feels comfortable in teaching these materials from a text may wish to choose the SIG-DP material as the text.

Further Study

The value of what is presented here is not only in the results of our efforts, but even more so in the process. We have shown how Illuminative Evaluation can be used with raters and how its results can help in the selection of one or the other set of materials depending on the goals of the customer.

The next round of this review might look at additional materials that could not be reviewed in the current round.

Bird Question and Answer Drill. Bird's Q and A drill is a computerized testbank and drill. It provides individuals with opportunities for self-evaluation and can be used effectively to identify areas needing further study. We asked the publisher for a complimentary evaluation copy so that these materials might be a part of this evaluation; the publisher declined the invitation.

Lockwood. *The CSP Review Manual* (1990) published by Lockwood Lyon and Kenniston Lord was acquired through the Association for Systems Management (ASM). It is a replacement for the *CSP Review Text* (1988) by Dr. Chadwick Nestman, CDP. The Lyon and Lord text could be used as another source when studying for the Systems Development specialty examination, but it does not correspond to the new ICCP examination structure.

Van Nostrand Reinhold has recently released the Lockwood Lyon *CDP Review Manual* (Fifth Edition). This manual does correspond to the new examination structure, and according to the publisher's announcement, this edition "encompasses all the material given in the new ICCP examinations". The timing of its release precluded it from being included in this evaluation.

Table 1: Members of DPMA's Certification Examination Review Material Evaluation Committee

Thomas H. Callan, CDP, CSP
Caterpillar, Inc.
600 W. Washington N1-AD210
East Peoria, IL 61630

Dr. Eli B. Cohen, CDP, CSP, CCP
College of Business
Eastern New Mexico University
Portales, NM 88130

Mark Cripe, CCP
Arnett Clinic
2250 Greenbush Street
Lafayette, IN 47904

Dr. Pal V. Rao, CDP
Library Services
Central Missouri State University
Warrenburg, MO 64093

Susan K. Rathman, CDP
2622 Innsbruck Trail
New Brighton, MN 55112

Wayne Troup
Central Maine Power
Edison Drive
Augusta, ME 04336

Table 2: Individual's Rating Form on Content

REVIEWER: DATE: REVIEW COURSE:

CORE EXAMINATION	ICCP GUIDE	Unacceptable 1	Less than Average 2	Average 3	Acceptable 4	Comprehensive 5
1. Human and Organization Framework	II.2					
2. Systems Concepts	II.3					
3. Data and Information	II.6					
4. Systems Development	II.7					
5. Technology	II.8					
6. Associated Disciplines	II.9					
MANAGEMENT						
1. General Management and Organizational Concepts	IV.2-4					
2. Project Management	IV.9-12					
3. Information Systems Management	IV.12-15					
SYSTEMS DEVELOPMENT						
1. Systems Analysis	V.2-4					
2. Systems Design and Implementation	V.5-10					
3. The Systems Analyst as a Professional	V.10-11					
PROCEDURAL PROGRAMMING						
1. Data and File Organization	III.2-3					
2. Program Design	III.3					
3. Procedural Program Structure	III.4					
4. Procedural Programming Considerations	III.5-6					
5. Integration with Hardware and Software	III.6-8					

Behavioral Anchors

Category	Meaning
Unacceptable	Provides no coverage
Less than average	Provides coverage only in passing, little depth, current or not
Average	Provides coverage of topics, few in depth, current
Acceptable	Provides coverage of 75% or more of topics, in depth, current
Comprehensive	Provides coverage of all constituent topics, in depth, current

Table 3: Individual's Rating Form on Goals Attainment and Bias

REVIEWER: DATE:

REVIEW COURSE:

	Not Offered 1	Offered OK 2	Offered Well Done 3
1. GOALS: INDIVIDUAL			
1.1 List of Objectives			
1.2 General Testing to determine deficiencies			
1.3 Pre/post test on each topic			
1.4 Visually appealing			
1.5 Glossary			
1.6 Index to other texts			
1.7 Ability to focus attention for review			
1.7.1 Headings			
1.7.2 Index			
1.7.3 Table of contents			
1.8 Suitable for quick review			
2. GOALS: GROUP			
2.1 Overhead Masters			
2.2 Instructor's Guide			
2.2.1 Teaching Outline			
2.2.2 Teaching Hints, Timing to expect to complete Assignments, etc.			
2.2.3 Instructions to set up class			
2.3 Other ancillary materials			
2.4 Pre/post tests			
2.5 Notes to each participant			

	YES	NO
3. RECOGNITION OF POTENTIAL BIAS		
3.1 Have you used the material in the past		
3.2 Did you help develop the material?		
3.3 Do you have any financial involvement with the material?		

Table 4: Summarized Ratings - Content for Bird and SIG-DP Materials

REVIEW MATERIAL COVERAGE EVALUATION SUMMARY

	<---BIRD---> PR	SR	VT	EC	MC	TC	BIRD AVG	<---SIGCP---> PR	SR	VT	EC	MC	TC	SIGCP AVG
CORE EXAMINATION														
1. HUMAN AND ORGANIZATION FRAMEWORK	3.0	3.0	4.8	NA	NA	NA	3.0	3.0	5.0	4.0	NA	NA	NA	4.0
2. SYSTEMS CONCEPTS	3.0	4.0	4.7	NA	NA	NA	3.5	3.0	5.0	4.0	NA	NA	NA	4.0
3. DATA AND INFORMATION	3.0	4.0	5.0	NA	NA	NA	3.5	3.0	5.0	4.0	NA	NA	NA	4.0
4. SYSTEMS DEVELOPMENT	3.0	4.0	4.7	NA	NA	NA	3.5	3.0	3.0	4.0	NA	NA	NA	3.5
5. TECHNOLOGY	3.0	3.0	4.0	NA	NA	NA	3.0	3.0	3.0	4.0	NA	NA	NA	3.3
6. ASSOCIATED DISCIPLINES	3.0	4.0	4.4	NA	NA	NA	3.5	3.0	1.0	1.0	NA	NA	NA	1.7
AVERAGE	3.0	3.7	4.6	NA	NA	NA	3.3	3.0	3.7	3.5	NA	NA	NA	3.4
SYSTEMS DEVELOPMENT														
1. SYSTEMS ANALYSIS	NA	NA	NA	5.0	NA	NA	5.0	NA	NA	NA	4.2	NA	NA	4.2
2. SYSTEMS DESIGN AND IMPLEMENTATION	NA	NA	NA	4.0	NA	NA	4.0	NA	NA	NA	1.8	NA	NA	1.8
3. THE SYSTEMS ANALYST	NA	NA	NA	2.5	NA	NA	2.5	NA	NA	NA	1.0	NA	NA	1.0
AVERAGE	NA	NA	NA	3.8	NA	NA	3.8	NA	NA	NA	2.3	NA	NA	2.3
MANAGEMENT														
1. GENERAL MANAGEMENT AND ORGANIZATIONAL CONCEPTS	NA	NA	NA	NA	NA	2.7	2.7	NA	NA	NA	NA	NA	4.1	4.1
2. PROJECT MANAGEMENT	NA	NA	NA	NA	NA	2.6	2.6	NA	NA	NA	NA	NA	2.8	2.8
3. INFORMATION SYSTEMS MANAGEMENT	NA	NA	NA	NA	NA	2.8	2.8	NA	NA	NA	NA	NA	3.1	3.1
AVERAGE	NA	NA	NA	NA	NA	2.7	2.7	NA	NA	NA	NA	NA	3.3	3.3
PROCEDURAL PROGRAMMING														
1. DATA AND FILE ORGANIZATION	NA	NA	NA	NA	2.0	NA	2.0	NA	NA	NA	NA	3.5	NA	3.5
2. PROGRAM DESIGN	NA	NA	NA	NA	2.0	NA	2.0	NA	NA	NA	NA	3.3	NA	3.3
3. PROCEDURAL PROGRAM STRUCTURE	NA	NA	NA	NA	2.0	NA	2.0	NA	NA	NA	NA	3.3	NA	3.3
4. PROCEDURAL PROGRAMMING CONSIDERATIONS	NA	NA	NA	NA	1.7	NA	1.7	NA	NA	NA	NA	2.6	NA	2.6
5. INTEGRATION WITH HARDWARE AND SOFTWARE	NA	NA	NA	NA	2.2	NA	2.2	NA	NA	NA	NA	3.4	NA	3.4
AVERAGE	NA	NA	NA	NA	2.0	NA	2.0	NA	NA	NA	NA	3.2	NA	3.2

Behavioral Anchors

	Category	Meaning
1	Unacceptable	Provides no coverage
2	Less than average	Provides coverage only in passing, little depth, current or not
3	Average	Provides coverage of topics, few in depth, current
4	Acceptable	Provides coverage of 75% or more of topics, in depth, current
5	Comprehensive	Provides coverage of all constituent topics, in depth, current

Table 5: Summarized Ratings - Goal Attainment and Bias

	<---BIRD---> PR	SR	VT	EC	MC	TC	BIRD AVG	<---SIGCP---> PR	SR	VT	EC	MC	TC	SIGCP AVG
1. GOALS: INDIVIDUAL														
1.1 List of Objectives	2	1	2	1	2	2	1.6	2	1	3	2	2	2	2.0
1.2 General Testing to determine deficiencies	1	1	3	1	2	1	1.5	1	1	2	*	1	1	1.2
1.3 Pre/post test on each topic	2	2	2	1	1	2	1.6	1	2	2	*	2	2	1.7
1.4 Visually appealing	3	3	3	3	2	3	2.8	1	2	3	3	2	2	2.2
1.5 Glossary	1	1	2	1	1	1	1.1	1	1	3	*	2	1	1.5
1.6 Index to other texts	2	1	3	2	1	2	1.8	2		2	2	2	3	2.8
1.7 Ability to focus attention for review														
1.7.1 Headings	2		3	1	1	3	2.0	2	2	3	3	2	3	2.5
1.7.2 Index	1	1	3	1	1	1	1.3	2	2	3	2	2	3	2.3
1.7.3 Table of contents	3	3	3	1	2	2	2.3	2	2	2	3	2	3	2.3
1.8 Suitable for quick review	2	2	3	1	2	3	2.2		1	2	3	2	2	2.0
2. GOALS: GROUP														
2.1 Overhead Masters	1	1	1	3	1	1	1.3	2	3	2	2	3	2	2.3
2.2 Instructor's Guide														
2.2.1 Teaching Outline	2	2	2	3	2	2	2.1		1	2	*	*	1	1.2
2.2.2 Teaching Hints, Timing to expect to complete Assignments, etc.	2	2	2	3	3	3	2.5	1	1	2	1	*	1	1.3
2.2.3 Instructions to set up class	3	3	2	3	3	3	2.8	1	1	2	*	*	1	1.2
2.3 Other ancillary materials	1	3	3	3	2	2	2.3		3	2	1	*	1	1.6
2.4 Pre/post tests	1	2	3	2	1	2	1.8	2	1	2	*	2	2	1.7
2.5 Notes to each participant	1	2	2	3	2	2	2.0	2	1	2	*	*	2	1.5
AVERAGE	1.8	1.9	2.5	1.9	1.7	2.1	1.9	1.6	1.6	2.3	1.7	1.9	1.9	1.8
3. RECOGNITION OF POTENTIAL BIAS														
3.1 Have you used the material in the past	N	N	N	N	N	N		N	Y		N		N	
3.2 Did you help develop the material?	N	N	N	N	N	N		N	Y*		N		N	
3.3 Do you have any financial involvement	N	N	N	N	N	N		N	Y		N		N	

* Only as it concerns SIG-CP

Behavioral Anchors

	Category	Meaning
1	Not Offered	Not offered as part of the material
2	Offered OK	Offered, but not exceptional
3	Offered Well Done	Offered and exceptional

THE CHANGING WORLD OF LANS: TCP/IP AND DVI

Dr. Greg Baur
Western Kentucky University

ABSTRACT

In recent years, advancements in technology have caused major changes in local area networks in their operation and use. Two of the most significant occurrences have been the development of TCP/IP and DVI. This paper will give an overview of both technologies and discuss the implications of the implementations of both.

OVERVIEW

In order to understand the impact of new technology, it is important to understand why the technology has been developed. What are the underlying reasons for this development? What problems are to be solved? Will new problems be the result of new technology implementation?

In this paper, we will examine answers to these questions relative to two technologies: TCP/IP (transmission control protocol/interconnect protocol) and DVI (digital video interactive). We will also give an overview of each technology and discuss applications. It is expected that the reader will have a basic understanding of data communication concepts.

TCP/IP

Data communications has become a fundamental part of computing. Sets of computers have been connected together using wide area, metropolitan area, and local area technologies, and the number of networks continues to grow by orders of magnitude on almost a daily basis.

Prior to the late 1980's, it was not possible for diverse computer systems to communicate with each other. This was because these systems had different hardware configurations and operating systems. The computing industry did not have, nor really desired, any kind of standardization.

In the late 1980's, a new technology was developed and known as internetworking or interneting. This technology permitted desperate physical networks and making them work as a connected unit. Interneting accommodates multiple diverse underlying hardware technologies by adding both physical connections and a new set of conventions. The internet technology hides the details of network hardware and permits computers to communicate independent of physical hardware connections.

During this time researchers at the Defence Advanced Research Project Agency (DARPA) studied the internet technology and developed the TCP/IP technology (named after the two standards it represents). The DARPA technology includes a set of network standards that satisfied the details of how computers communicate as well as a set of conventions for interconnecting networks and routing traffic.

To appreciate the power of the TCP/IP, it is necessary to understand the services it provides. The services defined by TCP/IP are standards or protocols, like TCP and IP, give the formulas for passing messages, specify the details of message formats and describe how to handle error conditions. These protocols are independent of any vendor's network hardware. In a sense, protocols are to communications as programming languages are to computation. A programming language allows a user to specify or understand a computation without knowing the details of the machine instruction set. Similarly, a communication protocol allows a user to specify or understand a communication without knowing the details of a particular vendor's network hardware.

From the user's point of view, a TCP/IP internet appears as a series of application programs designed to perform some useful communication task. The term interoperability is used to refer to the ability of diverse computing systems to cooperate in solving computational problems. Internet application programs exhibit a high degree of interoperability. Hence, it is not necessary for the user to understand TCP/IP to use these application programs. On the other hand, programmers who write the application programs need a clear understanding of the TCP/IP structure.

The most commonly used internet application services are electronic mail, file transfer, and remote login. Readers who have used other communications software are very familiar with these services.

As was pointed out above, a programmer who writes internet application programs views the internet much differently then the user. The programmer is interested in the services provided at the network level.

These services fall into one of two classifications. The first is known as connectionless packet delivery services. Connectionless delivery systems are an abstraction of the services offered by most packet switching networks. What this means is that a TCP/IP internet routes small messages from one machine to another based on address information contained in the message. The connectionless service sends each packet separately and so there is no guarantee of reliable in-order delivery at the destination. Connectionless packet delivery service is the basis of all internet services.

The second classification of service at the network level is reliable stream transport service. Most applications require more than packet delivery since they require automatic recovery from transmission errors, lost packets, or intermediate switch failure that slows the path between sender and receiver. These potential problems are handled by this service. It allows an application on one computer to set up a "connection" with an application on a second computer and pass large volumes of data between the two applications.

Many networks provide similar basic services. However, TCP/IP provides network technology independence to that it is not tied to a single vendor's hardware. TCP/IP specifies how data are to be broken up into datagrams and how the datagrams are to be transmitted across the connection.

TCP/IP also offers universal interconnection between any pair of senders and receivers. Each sender/receiver is assigned an address that is university known to all parties in the internet. Each datagram carries a source address and a destination address; these addresses are used for transmission error recovery and routing protocols, respectively.

Another service provided by TCP/IP is end-to-end acknowledgement. This provides communication between any source and destination where the source and destination may be two machines not connected to a common physical network.

It is a natural question to ask how two networks can interconnect with each other. Physically, two networks can only be connected by a computer that attaches to both networks. A simple attachment is insufficient, however, because there is no guarantee that the computer will cooperate with other machines that wish to communicate. To do this task requires that the computer be able to shift packets from one network to the other. Such a computer is often known as an internet gateway.

The task of interconnecting two networks with a gateway is not terribly complex. However, the task becomes more complex as more networks are added to the internet. For example, suppose that network n1 is connected to network n2 by gateway g1. Suppose also that n2 is connected to network n3 by gateway g2. Packets from n1 can go to either n2 or n3 and g1 must be able to recognize this.

It might seem that g1 must need a great deal of memory to know the locations of all machines on n2 and n3. The gateways, however, are usually microcomputers with little memory. The trick is that g1 stores only the address of the network for which the packet is destined. The destination network takes care of locating the machine.

This brief overview was intended to give the reader a global picture of TCP/IP. Details of TCP/IP implementation may be found in references at the end of this paper.

What the impact of TCP/IP on local are networks should now be clear. Machines on a local area network in a corporation can now communicate with machines on a different local area network in a second corporation. Machines on a local are network can also communicate with larger metropolitan or wide area networks.

As an example of the potential of TCP/IP, consider the following scenario. On a certain university campus, machines on a Novall network can communicate with machines on a Mac network or with machines on an Ethernet network. Prior to TCP/IP the same machine needed to be directly connected to each of the three networks to communicate with machines on those networks. The result is greater flexibility and cost saving.

DVI

The second significant technological development that has impacted LANs is the development of the DVI technology. Before beginning this discussion it is important f the reader to understand "the big picture" so that t. significance of this technology can be fully appreciated.

Since the beginning of the development and use of computing machines, there has been continuous major effort to improve the quality of techniques for the storage and retrieval of data. The data has traditionally been either text or numeric in nature and its manipulation was done on large machines.

In the late 1970's, the development of the personal computer (PC) showed that the power and capability of computing could be brought to the individual in a number of applications which were pretty much the same as those done on large machines. However, the PC development soon added graphics and sound to its capabilities and so the definition of data expanded to include those two media. While graphical data was acceptable to most users, sound data was not because of its variance from what was normally heard by humans. While acceptable, computer graphics suffered from the fact that "real" images could not be captured and stored by the computer.

Of course, movies and phonograph records had demonstrated that "real" voice and motion/still video data could be stored and retrieved. But, retrieval was slow and sequential in nature which was all right because humans know no better.

In the 1960's and 1970's, techniques for storage and retrieval of "real" voice and video improved with the development of the laser videodisc system. This system allowed random access to the voice and video data which was stored on a disk about the size of an LP record. The data on this medium was analog even though it was digitally encoded. Because the data was analog, it could not be manipulated by the computer and so the laser videodisc system was a separate entity from the computer. Would it ever be possible to digitize "real" voice and video data? If so, then it would be possible to combine the digitized voice and video with the power of the computer to produce an interactive multimedia system that would have applications in business, training, and education among others.

Today, computer support of human interaction with multimedia information is rapidly accelerating as evidenced by the phenomenal growth of digital technology in audio processing and the remarkable growth of computer graphics and video processing. This growth has been spurred by the development of compact disk - digital audio (CD-DA) in 1980 and the development of the compact disk read-only-memory (CD-ROM) technology in 1984. Both media use the digital encoding of digital data with high capacity storage. The CD-DA disk, 12 cm in diameter can hold up to 72 minutes of FM quality sound. The CD-ROM disk, approximately the same size, can hold over 600 megabytes of data including still pictures. It was also possible to store full motion video on the CD-ROM but it took all 600+ megabytes to store but 19 second of full motion video.

A major breakthrough to the storage of full motion video occurred in 1987 with the first demonstration of the DVI technology developed at the Sarnoff Research Center. The demonstration, using special boards in an IBM PC and a CD-ROM disk, showed that 72 minutes of full motion video could be stored on the 12 cm disk. The increased capability for storage of full motion video was made possible through the use of video compression and decompression techniques; these techniques will be explored later in this paper.

The final step in the brief historical journey occurred in 1989 when the DVI technology was sold to Intel, who is well known for its development of the 80 series (8088, 80286, 80386, etc.) CPU chips. Intel is vigorously working to make the DVI technology commercially feasible and is making good progress to that end. Initially, the DVI technology was being developed for the IBM PS/2 and PC XT/AT compatibles but will likely be extended to other vendors' products. Vendors other than Intel are also attempting to solve the full motion video problem but, so far, no significant solutions have been achieved.

When full motion video is achieved (and it will be) a real time interactive multimedia computer system will become available and soon be available to the general user. This system will have applications in business, training, and education and will make significant use of local area networks; more will be said about this later.

It might not be obvious as to why full motion video is a problem. The storage requirements for digital

video images is immense and is even worse for full motion video because of the number of images required.

Full motion video, to the human eye, must run at 30 frames (video images) per second. Recall from the earlier discussion that over 600 megabytes of storage was required for 19 seconds (870 still images) or approximately 689.5 kilobytes per image.

The transfer rate required to move such quantities of data to produce playback at the required 30 frames per second is impossible to achieve with current technology. Thus, storage requirements must be significantly reduced for the transfer of the full motion video data. This is the reason for the need for video compression.

The DVI technology has four unique components:

- a custom VLSI chip set
- a specification for a custom software interface
- video/audio data file formats
- compression and decompression algorithms

Hence it is clearly a combination of hardware and software components.

The VLSI chip set the heart of DVI technology. It consists of two VLSI chips and acts as a compressor. The speed of this chip set is what permits DVI to be able to accomplish its mission.

The functions of DVI are performed in application programs written in the C language. In some cases, assembly language is used. For users who are not C programmers, authoring systems are available. Specifications are given to the authoring system which generates the necessary DVI code.

Video compression and decompression algorithms have proved to be the big hang-up to the successful implementation of the DVI technology. A compression ratio of 200 to 1 is considered to be the minimum acceptable level compression. It has been very difficult to achieve this level of compression/decompression at the speed required to achieve full screen, full motion video playback. Lesser ratios have been achieved but the result has been full motion video that covers only 1/8, 1/4, or 1/2 of the screen. This lower level is acceptable for some applications but it is not accepted as an end by the developers of DVI.

Video compression algorithms, when working with either still or motion images, rely on redundancy and the human visual system. For an image that is in color, each picture element (pixel) has associated with it a bit string that contains color information. Two adjacent pixels that have the same color need store the color information only once. It is not difficult to realize that simple (less detail) video images are more easily compressed than complex video images.

The human vision system also plays a role in compression algorithm development. For example, there are a number of colors that are nearly indistinguishable to the human eye; color data for pixels that use these colors can be eliminated, thus aiding the compression problem and improving the compression ratio.

As yet, there have been no standards developed for video compression technology. Progress is being made to this end by the Instructional Organization for Standards (ISO), a subgroup, the Joint Photographic Experts Group (JPEG) is responsible for developing a standard for still image compression while another subgroup, the Moving Picture Coding Experts Group (MPEG) is responsible for developing the standard for motion image compression. The JPEG standard has had initial approval but the MPEG standard is far from completion because of the complexity of motion image compression. The DVI technology is as well developed as any other set of compression algorithms but it is proprietary and hence, details of their structure are not available.

The development of digitized audio and video has had and will continue to have a significant impact on LANs. It means that full multimedia capabilities can be shared by multiple workstations on a LAN and with TCP/IP, with multiple LANs.

There are a number of applications where the DVI and LAN technologies can work together. One is video conferencing. In this case, a meeting can be held with group members at remote sites. Participants can view and hear each other in real time and materials can be shared with other participants in real time. Time wasted in moving from and office to a control meeting site is eliminated.

A second set of applications for DVI and LANs is the area of education and training. Currently, to achieve full multimedia capability, a workstation for each individual must be equipment with additional peripherals

such as a CD-ROM drive. Such workstations have become known as multimedia personal computers (MPC). With DVI, one MPC can be designated as a server to a LAN with other workstations that do not have the peripherals. These remote workstations can share the content of one or more CD-ROM using the LAN, thus reducing the cost of each workstation.

SUMMARY

This paper has given an overview of two technologies TCP/IP and DVI. It has also discussed the rationale for the existence of better technologies and briefly examined possible applications that also utilize LANS.

BIBLIOGRAPHY

Ang, Peng et. al., "Video Compression Makes Big Gains," JEEE Spectrum, October 1191, pp. 16-19.

Comer, Douglas, Internetworking with TCP/IP Volume 1, Englewood Cliffs, NJ, Prentice Hall, 1991.

Luther, Aroh, Digital Video for the PC Environment, Second Edition, New York, McGraw Hill, 1991.

Madron, T.W., LANS: Applications of IEEE/ANSI 802 Standards, New York, John Wiley, 1989.

QUALITY, PRODUCTIVITY AND DP

by

Frank Greenwood
and
Jayanta K. Bandyopadhyay

Central Michigan University

Despite the billions invested in hardware and software, office productivity stagnates. DP professionals who know what to do about this will prosper. The first thing to remember is that our standard of living is driven by productivity; and, that productivity is driven by quality.

The more you produce for less, the higher your productivity and the better your standard of living. Our ability to improve our standard of living depends, in the long run, on our ability to raise output per worker. This presentation suggests how to increase office productivity.

AN ETHICAL LOOK AT UNETHICAL PRACTICES - A RANDOMIZED RESPONSE STUDY OF POSSIBLE STUDENT CHEATING ON GRADED PROGRAMMING ASSIGNMENTS

George W. Morgan, Southwest Texas St. Univ., San Marcos, Texas.
Walter E. Johnston, Southwest Texas St. Univ., San Marcos, Texas.
Mayur R. Mehta, Southwest Texas St. Univ., San marcos, Texas.

ABSTRACT

Most courses in computer information systems curricula use programming assignments as a means of teaching analytical and logical skills important in the development of computer solutions to business problems. Grades attained on these assignments are used as one criterion to measure student's performance and success in the course. A concern for instructors of such courses is the question of whether students are resorting to unethical practices to accomplish these assignments. This paper uses randomized response technique to examine this concern. The primary objective is to obtain a reliable estimate of how widespread such unethical practices are. This will provide insights into probable solutions to reduce the impact of such practices.

INTRODUCTION

Programming courses in Basic, Cobol, etc. are included in almost all computer information systems curricula. Commonly employed measures of student progress, performance and success are those grades attained on assigned 'programs' to be developed individually by the student. A continuing cause of concern for the instructor of such a course is the question of individual performance or effort. Is the program in fact the student's own work? If cheating does occur, how wide-spread is it?

Because of extremely sensitive (and, often, incriminating) subject matter, it would be difficult to elicit truthful responses from respondents by direct questioning or survey techniques. In addition, the reliability and validity of data collected would be suspect. Even when the researcher goes to the great length of assuring anonymity, subjects can be skeptical and may be reluctant to provide truthful responses. Direct survey techniques thus suffer from a major bias arising from deliberate falsification of information. Such systematic distortion jeopardize the validity of survey measurements since such response bias does not cancel out over repeated measurements.

The randomized response technique was posed initially by Warner (1965) as an alternative means of collecting data in such settings. An ethical survey technique, the randomized response technique is designed to elicit truthful responses to questions that may be embarrassing or even

incriminating for the respondent (Sudman and Bradburn, 1982 and Tracy and Fox, 1986). Here a sensitive question is presented in both the positive and the negative form. For example:

A. I have cheated on one or more programming assignments.
B. I have never cheated on a programming assignment.

Warner's technique directs the respondent to answer one or the other question depending on the outcome of a randomizing device (coin toss, shuffled deck of cards, spinners, etc.) that only the respondent sees. The response is no longer revealing, since no one except the respondent is aware of which question was answered. The interviewer is therefore in possession of the response ("yes" or "no") but does not know which form of the question was answered.

Elementary probability theory allows the researcher to obtain an estimate of proportion of subjects possessing the attribute in question (e.g. I have cheated on one or more programs) as:

$$\hat{\pi}_x = \frac{(\hat{\lambda} + p - 1)}{(2p - 1)} \qquad (1)$$

and

$$\textit{Estimated Var}(\hat{\pi}_x) = \frac{\pi(1-\pi)}{n} + \frac{p(1-p)}{n(2p-1)^2} \qquad (2)$$

where

p = known probability of "yes" response to sensitive question. $p \neq 0.5$
$\hat{\lambda}$ = observed sample proportion answering "yes".
π_x = estimated percent possessing attribute in sensitive question.

Still, the use of this related-question option seems to have provoked some degree of suspicion in respondents that some 'mathematical trickery' could be used to identify them after all. (Tracy and Fox, 1986). One alternative is to pair sensitive question with an innocuous or non-sensitive question. (Simmons, Horvitz, 1967)

In this study, we have employed one such model posed by Greenberg et al (1969, 1971, 1973, 1977) which was a drastic improvement over the model posed by Warner. In the unrelated-question model, one statement is embarrassing or sensitive and the second statement is non-sensitive. For example:

A. I have cheated on one or more programs.
B. I am a female student.

The proportion of subjects who had cheated on one or more programs (π_x) can be easily estimated as follows: (Tracy and Fox, 1986)

$$\hat{\pi}_x = \frac{[\hat{\lambda} - (1-p)\pi_y]}{p} \qquad (3)$$

with an

$$\textit{Estimated Var}(\pi_x) = \frac{\hat{\lambda}(1-\hat{\lambda})}{np^2} \qquad (4)$$

where

π_x = estimated percent cheating
π_y = percent of female subjects
p = known probability of answering sensitive question. **p** > 0.5,
$\hat{\lambda}$ = percent of "yes" responses

This model is most effective if the probability of a "yes" response to statement B (π_y) is known in advance!

Our primary research objective was to obtain a reliable estimate of the proportion of our (graduating) students who had in fact used unauthorized help on programming assignments during their coursework.

METHODOLOGY

Two sections of a senior-level computer information systems course (Data Management & Retrieval) agreed to participate in the randomized response study of the incidence of use of unauthorized assistance by students on programming assignments. A total of 35 students, all within 4 months of graduating, were involved.

Each student was asked to respond to one of the following two questions based on the outcome of a randomizing device.

1. I have received unauthorized help on one or more programming assignments while a student.

2. The month of my birth is January or February or March.

For our randomizing device, we used a standard deck of cards. We removed twelve red cards from a standard deck of 52 cards so that the altered deck included 26 black and 14 red cards. This was done to ensure that probability of response to sensitive statement was greater than 0.5 (i.e. **p** > 0.5). If a student drew a black card from a thoroughly shuffled deck of cards, the student was to respond to the sensitive statement. Otherwise, the student was to respond to the unrelated non-sensitive statement. Hence, 65% (26 black cards out of 40) of the responses were expected to be to the sensitive statement. Thus, the known probability of answering sensitive question (**p**) is 0.65,

Since the model works best if the proportion of "yes" responses to the non-sensitive statement (π_y) is known in advance, it was decided to use a statement pertaining to month of student's birth as the unrelated non-sensitive statement.

In order to describe the actual data collection procedure to the students, we stressed both the confidentiality of their responses and our need for accurate information. The deck of 40 cards used as the randomizing device was shown completely and the deck shuffled before each student in turn. The statements with instructions for indicating a response were fully described and distributed to each student.

RESULTS AND FINDINGS

Table I presents the detailed

breakdown of data obtained. Of the thirty-five students who participated in the study, a total of eight "yes" responses were obtained.

TABLE I

SECT	# IN CLASS	# of YES Responses	# with 1st Qtr Birthday
1	17	5 (λ_1 = 5/17)	6 (π_{y1} = 6/17)
2	18	3 (λ_2 = 3/18)	2 (π_{y2} = 2/18)
TOT	35	8 (λ = 8/35)	8 (π_y = 8/35)

From vita sheets prepared much earlier in the course it was known that eight of the thirty-five students were born in the first three months of the year (π_y = 8/35 = 0.2287).

From equation (3), the estimated percent cheating based on both sections combined is:

$$\pi_x = [(8/35) - (1.65)(0.2287)]/.65 = 0.2287$$

with an estimated standard error of (from equation (4)):

Estimated Standard Error (π_x) =

$$\sqrt{(0.2287*(1-0.22877)/(35*(0.65)^2} = 0.1092$$

Similar estimates for each section are presented in Table II.

We believe this estimate of 23% to be a reasonably accurate (free of response bias) estimate of cheating by students who have progressed as far as their senior year in college.

TABLE II

SECT	ESTIMATED $\hat{\pi}_x$	ESTIMATED STD ERR ($\hat{\pi}_x$)
1	0.2624	0.1700
2	0.1966	0.1351
TOTAL	0.2287	0.1092

We thus estimate that as many as 45% (π_x + 2 std. errors) of those students who are graduating will have used unauthorized assistance on graded programs. Estimates for individual sections also show similar results, with upper limits ranging from 47% to 60%. Since a relatively high percentage of students will have used unethical means for accomplishing programming assignments, there seems to be strong evidence for having such assignments carry only a minimal percentage of the final course grade.

Obviously, the study suffers from the limitations of using a small sample. Further studies should probably be conducted on a larger body of students in the latter part of their studies in the CIS program using the randomized response technique or a variant such as the covariate randomized response model (Scheers and Dayton, 1988).

REFERENCES

Fox, A. J., and Tracy, P. E. (1986), Randomized Response: A Method for Sensitive Surveys, Beverly Hills: Sage.

Greenberg, B. G., Abul-Ela, A., Simmons, W. R., and Horvitz, D. G. (1969), "The Unrelated Question Randomized Response Model: Theoretical Framework," Journal of the American Statistical Association, 64, 520-539.

Scheers, N. J., and Dayton, C. M. (1986), "RRCOV: Computer Program for Covariate Randomized Response Models," The American Statistician, 40, 229.

Sudman, S., and Bradburn, N. M. (1974), Response Effects in Surveys, Chicago: Aldine.

Tracy, P. E., and Fox, J. A. (1981), "The Validity of Randomized Response for Sensitive Measurements," American Sociological Review, 46: 187-200.

Warner, S. L. (1965), "Randomized Response: A Survey Technique for Eliminating Evasive Answer Bias," Journal of the American Statistical Association, 60,63-69.

HANDS-ON TESTING FOR COMPUTER LITERACY

Marty McClelland, School of Business, North Carolina Central University, Durham, NC 27707 919-560-6457

ABSTRACT

The objective of a computer literacy course is for students to gain proficiency in computer skills. After unsatisfactory experiences using paper and pencil exams, I have experimented with the use of hands-on tests at the computer to assess the students proficiency with various computer skills. Students accept the hands-on tests as a fair assessment of their competency and seem motivated to prepare and practice their skills in anticipation of the test.

INTRODUCTION

When I first started teaching a computer literacy course covering DOS, WordPerfect, Lotus, and DBase III, I used the multiple choice questions and short answer questions in the instructors manual to evaluate the students. The students did not feel that their scores on the tests reflected what they had learned, and I agreed with them. In striving to find another way to assess student proficiency, I started using Hands-On testing in addition to written tests. Generally, I use Hand-On tests for assessing the fundamental skills and the written tests for definitions, theory, and concepts.

I give students a list of learning objectives that the lab practical will cover. Students seem more motivated to master their fundamental skills since they know they are going to personally demonstrate their competency to me. Students will spend time practicing before their Hands-On test and they will ask questions about how to do a particular task. In class, I may use structured exercises to go over learning objectives. For example, I may have students work in small groups for ten minutes at the end of class and write down the command or sequence of commands to do each learning objective. I bring a portable computer to the classroom so students can use it to verify their command sequence.

Another advantage of the Hands-On test is that homework remains in the role of providing students with learning opportunities with less emphasis on grading. Even if students miss turning in a homework assignment, they recognize the need for learning the skills covered in the assignment. I am comfortable with students helping each other on the assignments or completing the assignment in pairs since the Hands-On test will assess individual competency.

PROCEDURE FOR HANDS-ON TEST

I give students a list of skills or learning objectives well before the scheduled time for the Hands-On test. Figure 1 is the list of learning objectives for DOS. During the week before the Hands-On test, I will take class time for students to review the list of learning objectives and answer questions relating to the learning objectives. I will also point out if there are any learning objectives that will not be included in the Hands-On test. For example, I usually use a written test for assessing understanding of WordPerfect merge procedures, so I point out that merge related learning objectives will be on the written test and not part of the Hands-On test.

Students sign up for a test time. I can effectively handle four to five students at a time. I usually plan 20 minutes for each test in DOS and WordPerfect, and I allow 30 to 40 minutes for LOTUS and Dbase. However, I do allow students to take extra time if needed. Thus, I usually have two extra testing stations set up just in case. I also plan in slack

time after about 90 minutes of testing.

In the following sections, I describe the Hands-On test for DOS as an illustration of the Hands-On tests given for each module of the computer literacy course: DOS, WordPerfect, Lotus, and DBase.
The administration of the Hands-On test is exhausting. However, the grading is complete when the Hands-On testing is complete. For me, the tradeoff of spending a long time administering the test is balanced by a short amount of time required to review the scoring and record grades.

DOS HANDS-ON TEST

Test Administration

I prepare disks like the examples in Figure 2 for each testing station. An easy way to create dummy files is to use the *DOS* copy command where the source file is what is typed at the console until a ctrl-z is typed. For example, to create a file TEST.DOC on the B disk type:

```
COPY CON B:TEST.DOC
dummy file for DOS test
^z
```

The file TEST.DOC contains one line of text. The data disk can be write protected to avoid corruption during the testing.

Put one copy of the test instructions, Figure 3, at each testing station. As student's arrive, I have them put their name on their score sheet, Figure 4, and return it to me. I remind student's that they do not have to wait for me to check every command, but to make sure I have checked their work before it scrolls off the screen. I score the students as they work, and when possible I have them save their test files as documentation of their test. The score sheet in Figure 4 shows the guidelines I use for grading. The most efficient correct command sequence is a score of 10. If all the commands are correct, but the student has used multiple commands then the score is a 9. If the student makes several incorrect commands before giving the correct command, the score is 7. The student can ask for a tip or hint; then the score is a 5 if the resulting command is correct. If the student can not execute the command, the score is 0.

Test Documentation

Each student receives the completed score sheet in Figure 4. After correct completion of the Hands-On test, the system disk will contain a directory under the students last name. This directory contains the README.TXT file. Other files copied to the system disk during the test have been deleted. The data disk is unchanged.

I consolidate all the files and directories on test disks to one disk for the class as documentation in case of questions regarding the test. I use the following command to selectively copy files and directories.

```
XCOPY A:*.* B:/E/P/S
```

SUMMARY

I plan to continue using Hands-On testing. By using the Hands-On testing, I am satisfied that all students who pass the course have at least a rudimentary skill level in each module. In addition, students seem more motivated to do the homework and to practice their computer skills because they "are going to need to do it on the test".

NOTE: The paper was condensed to meet proceedings limitations. A version of the paper which includes examples of the Hands-On test for each module (DOS, WordPerfect, Lotus, and DBase) is available from the author.

DOS Learning Objectives for Hands-On Test

Sign up for a time slot. You will have 15 minutes to demonstrate your competency in DOS. For the demonstration, you may use your DOS reference index card.

recognize when computer is at DOS operating level

use DOS to:
- change default drive
- format data disk
- format system disk
- remove a single file
- remove a group of files
- duplicate files
- list filenames of files on disk or in directory
- change the name of a file
- display ASCII (or DOS text) files on the screen
- send DOS command output to printer
- create subdirectories
- remove subdirectories
- access subdirectories
- identify subdirectories
- identify autoexec.bat, config.sys, and path command
- clear the screen

for filenames
- access files using path, drive, name, and extension
- use wildcard characters

distinguish between DOS utility programs and DOS commands

FIGURE 1
DOS LEARNING OBJECTIVES

Disks used for DOS Hands-On Test

system disk is in A drive

```
COMMAND.COM
```

data disk is in B drive

```
TEST.DOC
EXAM.TXT
HOMEWORK.DOC
README.TXT
```

FIGURE 2
DISK CONTENTS FOR DOS TEST

DOS Lab Test Instructions

Make sure I have scored your work before the commands scroll off the screen.

The use of a one line command will receive a higher score than the use of multiple commands to perform the same action.

You can ask for help, but this will lower your score.

Put the systems disk in the A drive and the data disk in the B drive. Proceed to the steps below after the screen shows A>

1. Show the directory listing for the B drive data disk.

2. On the A drive disk create two subdirectories:
   - use CIS2700 as the name of one subdirectory

3. - use your-last-name (ex:PROFESSO is directory name for Professor) for the other directory name

_____ HAVE YOUR WORK CHECKED BEFORE IT SCROLLS OFF THE SCREEN

4. Make the your-last-name directory the current directory.

5. Copy all the files from the data disk in the B drive to the your-last-name directory on the A disk.

   Remove the disk from the B drive.

6. Change the name of EXAM.TXT to 2700EXAM.DOC Show the directory listing with the new file name.

7. Display contents of README.TXT on the screen

8. Remove the files TEST.DOC, HOMEWORK.DOC, and 2700EXAM.DOC Show the directory listing without the files.

9. Remove the directory CIS2700
   Show the directory listing for the A drive disk

_____ CHECK BY INSTRUCTOR NEEDED BEFORE CONTINUING

10. Give the command to clear the screen.

_____ CHECK BY INSTRUCTOR NEEDED

FIGURE 3
INSTRUCTIONS TO STUDENT FOR DOS TEST

DOS Lab Test

print name _____

The use of a one line command will receive a higher score than the use of multiple commands to perform the same action.

You can ask for help, but this will lower your score.

____ 1. Show the directory listing for the B drive data disk.

____ 2. Set the prompt to show the current directory.

____ 3. On the A drive disk create two subdirectories:
- use CIS2700 as the name of one subdirectory
- use your-last-name (ex:PROFESSO is directory name for Professor) for the other directory name

____ 4. Make the your-last-name directory the current directory.

____ 5. Copy all the files from the data disk in the B drive to the your-last-name directory on the A disk.

Remove the disk from the B drive.

____ 6. Change the name of EXAM.TXT to 2700EXAM.DOC Show the directory listing with the new file name.

____ 7. Display contents of README.TXT on the screen

____ 8. Remove the files TEST.DOC, HOMEWORK.DOC, and 2700EXAM.DOC Show the directory listing without the files.

____ 9. Remove the directory CIS2700
Show the directory listing for the A drive disk

____ 10. Give the command to clear the screen.

_____ total score

guidelines
10 single line command
9 multiple correct commands
7 multiple attempts, but action is completed
5 hint for syntax
0 does not know command

FIGURE 4
INSTRUCTOR SCORE SHEET FOR DOS TEST

The Use of the Myers Briggs to Improve Novice Programmers' Problem Solving Abilities

Catherine Bishop-Clark
Miami University, Ohio

The Myers Briggs Type Indicator (MBTI) is based on Carl Jung's theory of personality and is currently one of the most popular personality assessment instruments. The MBTI points to our preference in dealing with the inner world or the outer world, in perceiving information, in making decisions, and our style when organizing the outer world. These preferences have a powerful impact on our problem solving process, including the process used to write computer programs. Computer educators can use knowledge of type to design more effective class sessions, to help students become more aware of their particular style, and to show students how style influences the programs they write and the problems they solve.

Introduction

This paper reviews how the personality dimensions measured by the Myers Briggs Type Indicator relate to problem solving and how educators in the computer field can use knowledge of type to help their students become better problem solvers. The MBTI is an instrument designed to measure four dimensions of an individual's personality: introversion/extroversion, sensing/intuitive, thinking/feeling, judgment/perception. Each of the four scales represents two opposite preferences. For every pair of items an individual prefers one pole over the other.

While there is an abundance of information on using the MBTI in counseling, education, career guidance, and teamwork there is significantly less on how type influences problem solving. Computerized searches and searches of citations yielded a total of four papers that related MBTI to computer programming (Carland and Carland 1990, Coreman 1986, Evans and Simkin 1989, Werth 1986). With the exception of the Carlands' paper, the remainder of the papers used type as a predictor of success in computer programming classes. Carlands' (1990) paper discussed the profile of a business information systems class and the teaching implications of that particular profile.

This paper is organized in three sections. The first section provides an introduction to type theory, including its history and basic components. The second section discusses how type is related to problem solving and the third section provides a personality profile of a typical introductory programming class and discusses the ways in which the professor can use knowledge of type to help the students become better problem solvers.

Introduction to Type Theory

History

In one of his classic books Psychological Types (1923), Carl Jung (a Swiss psychiatrist) described the systematic ways in which people differ. He focused heavily on the attitude which a person faced the environment, and he believed that people tended to be either introverted or extroverted in their thought processes. Jung believed that conscious mental activities could be described as being one of four categories: sensing, intuitive, thinking, and feeling. The first two activities (sensing and intuitive) were the two general approaches to how individuals perceive information and hence named the perceiving dimension and the second two categories (thinking and feeling) were the general approaches to how individuals make decisions and was described as the judging dimension.

The MBTI was developed in the early 1950s by Katherine Cook Briggs and Isabell Briggs Myers. The indicator was designed to make Jung's theory more explicit and practical in everyday lives. They developed a series of self-reported items which helped people determine their psychological type. In 1975, Consulting Psychologists Press published the MBTI. Since that time the use of the MBTI has continued to rise in education, counseling, business, government and religious communities (McCaulley, 1987).

Dimensions of the MBTI

The MBTI measures preferences on four dimensions, each having two poles. Hirsh and Kummerow (1989) describe the four dimensions as energizing, attending, deciding, and living.

The first category (energizing) refers to a persons orientation toward the world. The two poles of the

category are introvert and extrovert. Although the common description of an introvert is "shy" and extrovert is "outgoing", the dimension is intended to measure how a person is energized. Extraversion (E) describes an attitude where attention is drawn out toward objects and people. Extroverts tend to draw energy from the external world of people and things. They prefer to communicate more by talking and often process information aloud. Introversion (I) describes an attitude where attention is drawn toward the inner world of ideas. Introverts tend to draw energy from the internal world of ideas, emotions, and impressions. They tend to process information inside their head. Whereas, extroverts often act without thinking, introverts often think without acting.

The second category refers to how a person perceives information. The two poles of the category are sensing (S) and intuitive (I). A sensing person tends to perceive observable facts through the five senses. An intuitive person perceives information based on the meanings, relationships or possibilities beyond the information gathered from one's senses. Sensors are often described as being more practical; whereas intuitors are described as more innovative.

The third dimension is referred to as the judging dimension and refers to how a person to makes decisions. The two poles of this dimension are feeling (F) and thinking (T). Feelers tend to be very attuned to their own feelings and the feelings of others. They base their decisions on what is important to themselves and others. On the other hand, thinkers base their decisions on an objective, impersonal and logical analysis. They are often focused on cause/effect relationships and seek an objective standard of the truth.

The last dimension of the MBTI refers to how one is oriented toward the outer world. The two poles of this dimensions are judgement and perception. Judgers are people who prefer to work in a linear, orderly method. They seek closure, tend to be organized and want things settled. Those who prefer a perceptive process would rather live a flexible spontaneous life. They prefer to keep their options open and are often viewed as spontaneous.

The 16 Types

The four dimensions (each with two poles) can be combined to identify 16 different types. The type is identified with four letters such as ISTJ (Introverted, Sensing, Thinking, Judging). It is important to highlight the fact that no preference is better than another and that we actually use all preferences at one time or another. Type does not explain everything and does not measure abilities. The MBTI simply points to our preference in dealing with the inner world or the outer world (I/E), perceiving information (S/I), making decisions (T/F), and organizing the outer world (J/P).

Dominant/Inferior Type

As mentioned above, the mental or cognitive dimensions of the MBTI are the perceiving (S/N) and the judging (T/F) dimensions. The theory holds that these four preferences (S/N/T/F) are rank ordered with the most preferred being the dominant, the second most preferred the auxiliary, the third the tertiary; and the least preferred, the inferior. The theory maintains that the dominant will become the most developed and hence the best process. The inferior on the other hand will be the least preferred and the least developed, hence the weakest personality dimension (Myers and McCaulley, 1985). The process of identifying the rank order of types is described in detail by Myers and McCaulley (1989). For the purposes of this paper it suffices to say that each person uses all four processes but not at the same level. The dominant function is the most developed and our "best" function. The inferior function is the least developed and the most likely to be overlooked.

MBTI and Problem Solving

Type and its relationship to problem solving

The individual differences in the way people perceive information (S/I) and make decisions (F/T) leads to different problem solving styles. Since the perceiving and the judging dimensions are considered the cognitive dimensions of Jung's theory, they play the greater role in problem solving (Yokomoto and Ware, 1982; McCaulley, 1987; Hellriegel and Slocum, 1975).

Sensing/Intuitive

Yokomoto and Ware (1982) consider the perceiving dimension (sensing/intuitive) the most relevant to problem solving. Sensors focus on facts and immediate realities, often see the trees and not the forest, tend to move from specific to general (inductive), and are very oriented to the present. Intuitors are good at seeing new possibilities, good at looking at different way of viewing a problem,

see the implications of the big picture, tend to move from the general to the specific (deductive), and are future oriented.

Thinking/Feeling

Also playing a very important role in the problem solving realm is the judging dimension (thinking and feeling). As previously described the judging dimension is most often described as our method of making decisions. A person with a preference toward thinking is very analytical and objective, concerned with definitions and connections, and is often very impersonal in their analysis. The feeling person is more likely to place emphasis on how deeply they care about the gains or losses of an alternative as well as what others will feel about the solution.

Introverted/Extroverted/Judging/Perceptive

While the cognitive functions appear to play the more important role with respect to problem solving, the four attitudes also influence ones approach to solving problems. An introvert will place greater weight on inner ideas and concepts. Introverts are more likely to need quite concentration and have more of a need to work without interruptions (Hellriegel and Slocum, 1975). When solving problems, extroverts place greater weight views of others and are more likely to discuss and communicate during the problem solving process. Judgers are concerned with organizing and structuring their problem solving process and are eager to seek closure. Perceptive types will be much more adaptable, curious and spontaneous in solving problems.

The dimensions of type interact to create a unique but systematic style of problem solving. To illustrate an ISTJ will approach and solve a problem very differently than an ENFP. An ISTJ wants a clear idea of the problem, approaches it by looking at the facts and uses a logical, impersonal, step-by-step approach. ISTJs are organized and seek closure. On the opposite end of the spectrum, an ENFP will entertain all sorts of possibilities, seek feedback from the environment, and emphasize the human aspects of the situation. ENFPs find the ISTJs approach slow and unimaginative. ISTJs find the ENFPs approach haphazard and inefficient.

Using Knowledge of Type to Improve Students Programming Skills

In Spring of 1991, the MBTI was administered to two sections of an introductory programming class (n=39). Form G of MBTI was administered the last 20 minutes of the first day of class. The students were informed of their group profiles (the class averages) as well as their individual profiles. The intended use of the MBTI was to get a personality profile of the class and to use this profile to improve teaching. As the class evolved, a secondary use became to highlight how personality affected an individual's approach to problem solving. Throughout the course, we discussed how individual profiles influence approaches to solving computer programs. The class met 3 hours a week for 16 weeks and the programming language used was BASIC.

The primary idea behind using type to improve student's problem solving skills is that by increasing the student's awareness of their own strengths and weaknesses as problem solvers, students can deliberately focus on the problematic areas. Additionally, if the instructor is aware of a class profile, they are able to use this information to design more effective class sessions.

Introversion Extroversion

In many ways, programming is an introverted activity. When ranking over 160 occupations with regard to the percent extroverts in the field, computer programming was one of the last 10 occupations. In other words, a very large percentage (60%) of the people who choose careers in programming are introverts (Myers and McCaulley, 1989, pg 246). The classes involved in this study tested to be 56% introverts. Werth (1986) found a similar trend. She reported that 63% of a introductory programming class tested as introverted. This is an especially high percentage of introverts when considering the national average is 25% introverts and 75% extroverts (Myers and McCaulley, 1989).

The fact that introverts may be over represented in programming classes has implications for the professor and the students. The professor needs to be alerted to the fact that one of the characteristics of introverts is that they prefer to think before speaking. Introverts often learn best when quietly studying a particular topic individually. Instead of asking for immediate feedback, an instructors of a largely introverted class should be sure to give plenty of wait time after posing a question. To assure participation from both introverts and extroverts, ask a question, ask that no one answer the question for a minute

or so, and then ask for participation from the group. An informal evaluation of this technique proved that the technique was quite successful and given a few minutes wait time, the introverted students were far more likely to participate.

As far as the problem solving process goes, introverts need to be aware that they are likely to work best independently and they have a tendency to work alone. Instructors can help students understand that computer programming courses are designed for introverted students but "real world" programming requires a different set of strengths. In introductory programming classes, programs are for the most part individually assigned and students receive severe penalties for extensive collaboration. While introverts may be more successful in a computer programming class, students should be alerted to the fact that in "real jobs" interaction is an absolute necessary. Jung's theory would suggest that highly introverted programmers would be especially weak in seeking feedback from the external environment when solving and designing computer solutions. This may partially explain why computer groups are notorious for delivering programs and systems that do not meet customer specifications.

Sensing/Intuitive

The sensing/intuitive dimension is the one dimension that seems to have the least obvious impact on the specific skill of computer programming. On one hand, computer programming is a very hands-on and practical activity. One runs a program and the results of the run are tangible. On the other hand developing programs require that one represent concrete everyday situations in abstract forms. In that sense computer programming requires an intuitive leap in order to map what happens in a computer program to the reality it represents. It may be that the process of designing and interpreting written code is more intuitive; whereas, the process of debugging is more sensing.

The instructor can alert students to some of the findings of previous studies. For instance the instructor can share Yokomoto and Ware's (1982) findings that sensing students tend to dismiss concepts; whereas, intuitive types are more likely to dismiss the specific routines. To illustrate, in this programming class we discussed and illustrated several sorts. According to type theory, sensors are more likely to concentrate on code and intuitives more likely to concentrate on the general concept. By alerting students to their bias, they may become more aware of the information they are more likely to dismiss because of their problem solving style. Additionally, the instructor may find it beneficial to group sensors with intuitors to study and discuss sorting at both the conceptual and the coding level.

The profile of the class may give the instructor a general idea of what the preferred and more effective examples may be. In the class involved in this study 59% of the class tested as sensors; hence the majority of the class prefered practical and tangible examples.

Thinking/Feeling

One of the problems that novice programmers experience is they sometimes give meaning to data or variable names that should be approached as meaningless. For instance if a variable is named minimum, the student will assume the variable will automatically take on the minimum value. This may be especially a problem with feeling type students. By alerting F type students that this is a difficulty they may encounter they can be more deliberate in exercising the thinking function. We can also use the thinking dimension to alert those students that are strong T's that not everyone's thought process (especially their future clients) will be as logical as their own.

In this particular class 67% of the class was thinking. One of the interesting findings in this class is that feeling types were more likely to drop the class than thinking types (46% feeling types dropped the class while only 18% of thinking types dropped the class). Although the data reported cannot be generalized, the finding makes intuitive sense. Computer programming requires an impersonal, sequential, and logical analysis (the exact description of thinking in the MBTI). While feeling has a place in the design of user interfaces and in dealing with people who will use the system, it has no place in the actual programming of a machine. Feeling students are more likely to be quite sensitive to relationships with the teacher. They are likely to be especially motivated by encouragement or complements by the professor.

Judging / Perceptive

Research has indicated that some of the weakest phases of a beginning programmers problem solving process is in the problem representation and in the

design phase (Allwood 1986). Students immediately jump into writing code without any advanced planning. Many introductory classes utilize tools (flowcharts, structure charts, pseudocode) that are meant to assist students in their design endeavors. Instructors frequently require the use of a tool and ask that the student complete the design (via the tool) before beginning code (a procedure much more appealing to the judging personality).

Type theory suggests that judging type students may be more successful and certainly enjoy the use of planning tools more so than their perceiving counterparts. Type theory predicts that perceiving students need additional help and deliberate activities to exercise the planning activity. On the other hand, students who have a strong J type need to be alerted because they have such a strong preference for structure and closure they will have a tendency to seek closure too soon. In other words, strong J types may have a tendency to ignore relevant information if it becomes apparent after they have designed an appropriate solution.

Other Related Benefits

Although the focus of this section has been to highlight how type can be used to improve problem solving in general, knowledge of type has some added benefits to students who will some day be professional programmers and analysts. Through the process of increasing a student awareness of their particular problem solving style, students begin to acknowledge and even appreciate differences among their fellow students. By learning more about their individual problem solving style students were sensitized to differences among problems solvers and became more sensitive to issues involving cooperation and collaboration - issues that tend to be overlooked in an already tightly packed introductory programming class.

References

Allwood (1986). Novices on the Computer: a review of the literature. International Journal of Man-Machine Studies, 25, 633-658.

Boreham, N.C. (1987). Causal attribution by sensing and intuitive types during diagnositc problem solving. Instructional Science, 16, 123-136.

Carland, J.A. & Carland, J.W. (1990). Cognitive Styles and the Education of Computer Information Systems Students. Journal of Research on Computing in Education, 23, 114-126.

Coreman, L. (1986). Cognitive Style, Personality Type, and Learning Ability as Factors in Predicting the Success of the Beginning Programming Students. SIGCSE Bulletin, 18(4), 80-89.

Evans, G. and Simkin, M. "What Best Predict Computer Proficiency. Communications of the ACM, 32, 1322-1327.

Hellriegel, D. & Slocum, J.W. (1975). Managerial Problem-solving Styles. Business Horizons, 18 (December), 29-37.

Hirsh, S. & Kummerow J. (1989). LifeTypes New York: Warner Communication Company.

Jung, C. (1923). Psychological Types New York: Harcourt, Brace & Co. Inc.

Keirsey, D. & Bates, M. (1984). Please Understand Me: Character & Temperament Types (5th Ed.) Del Mar, CA: Prometheus Nemesis Book Company.

Lawrence, G. (1982). People Types and Tiger Stripes: A Practical Guide to Learning Styles (2nd Ed.) Gainesville: Center for Applications of Psychological Type Inc.

McCaulley, M.H. (1987). The Myers-Briggs Type Indicator: A Jungian Model for Problem Solving. New Directions for Teaching and Learning, 30, 37-53.

Myers, I.B. (1980) Introduction to Type (4th ed.) Palo Alto, Calif.: Consulting Psychologist Press.
Myers, I.B., & McCaulley, M.H. (1989) Manual: A Guide to Development and Use of the Myers-Briggs Type Indicator. Palo Alto, Calif.: Consulting Psychologists Press.

Werth, L. (1986) "Predicting Student Performance in Beginning Computer Science Class." Proceedings of SIGSCE Conference. 138-143.

Yokomoto, C.F. & Ware, J.R. (1982) "Improving Problem Solving Performance Using the MBTI." In L.P. Graysona nd J.M. Biedenbach (eds.), Proceedings of the 90th Annual Conference of the Americal Society for Engineering Education.

A Comprehensive Survey of USA Two-Year Academic Undergraduate Programs in Computer Information Systems

Herbert E. Longenecker, Jr.(1), David L. Feinstein(1),Robert Fournier(2), Daniel Claborn(3) and William R. Reaugh(4)
(1) University of South Alabama, Mobile, Alabama 36688 ; (2) Alpena Community College, Alpena, Michigan 49707 ; (3) Oklahoma State University, Okmulgee, Oklahoma 74447; (4) Caterpillar Incorporated, E. Peoria, Illinois 61630

Abstract

A survey was conducted of academic programs in Information Systems at two-year institutions to provide the background for the development of a new curriculum model. Five hundred programs in Information Systems at two-year institutions were surveyed. The response rate was twenty-four per cent with representation from thirty-five states. The results suggest that the programs are designed for students to complete an associate degree or to provide the opportunity for persons already in the field to improve or upgrade their skills. The response indicated a close agreement with the knowledge clusters proposed in the recently completed four year curriculum, IS'90. This suggests that much of the work done for IS'90 may be used to form the basis for the two-year curriculum. It also suggests that articulation between two and four year programs in IS may be developed with a minimum of lost time for the students.

Introduction

The Data Processing Management Association (DPMA) has sponsored curriculum development projects in Information Systems (IS). Previous efforts have resulted in the development of Model Curricula for Information Systems for both four year (DPMA 1981,6; Longenecker and Feinstein 1991 a,b,c,d) and two-year curricula (DPMA 1985). The Association for Computing Machinery (ACM) has also been active in IS curriculum development effort for four year programs (ACM 1981,2,3; Ashenhurst 1972; Couger 1973), and is currently involved in developing two-year curriculum models.

This survey was developed by the DPMA sponsored Two-year Curriculum Committee to gather information to lay the framework for the model. The survey was based on the work of the four year committee (Longenecker and Feinstein 1991 a,b,c,d) with modifications suggested by the Two-year Committee. An attempt was made to incorporate concerns expressed by the IS profession (Hoffman 1991; Mackowiak 1992).

It was felt that the operating environment and exit expectations for two-year students was significantly different from that of the four year institutions. Thus, elements of the survey were developed to address these factors including:

- student background and preparation,
- career objectives,
- program responsiveness to industry needs,
- variations in length of program,
- courses used by non-traditional students to upgrade their skills
- articulation with four year programs.

Definition of the Sample

Peterson's Guide to Colleges and Schools provided a list of 1000 appropriate two-year programs. Five hundred were randomly selected for the survey. The survey was distributed through regular mail in January 1992. A reminder post card was sent in February to improve the response rate. Forty additional responses were received after the second mailing. By May 1, 1992, 120 programs had responded giving a 24% sampling rate.

Program Included in Survey Sample

Programs in Computer Information Systems, Computer Science, and Data Processing were selected from titles from the Peterson's Guide. Base on the work done for IS'90, it was anticipated that many of the programs with titles other than Information Systems would have course work closely resembling those in IS. Peterson's Guide supplied the names of department heads. Results from the survey indicate considerable turnover for the department heads. The survey also suggested a trend to rename programs from "Data Processing" to "Computer Information Systems" with eleven out of the one hundred and twenty making this

change.

The survey was sent to programs in 50 states with responses back from 35.

Survey Metrics and Processing

A five point Likert scale was used to sample opinion. Upon processing responses were converted from a -100 to +100 score. Zero was Neutral, 50 was somewhat agree, etc. The survey had 231 questions. Only summary results are presented in this paper.

Curriculum Document Mission

It was desire of the Committee to provide the academic community with as useful a curriculum document as possible. The survey therefore asked the respondents to identify how such a document could be used. The results indicate that it should assist in

- Identifying courses and defining their content
- Specifying depth of coverage and behavioral testing methodology
- Sequencing courses and structuring the overall curriculum to provide step-wise learning to accomplish objectives and to ensure retention
- Anticipating and accommodating continual curriculum change to reflect the rapidly changing IS environment
- Graduating students who are competent, confident and ethical in their chosen IS environment

Two-year Program Operating Constraints

In developing the curriculum model for two-year institutions for Information Systems, several factors had to be considered. It was found that students should be provided with a path into four-year programs. Of equivalent importance are (1) the associates degree as a terminal degree and (2) courses to support the "End User Specialist" who is being looked at to provide computing support in many organizations for both hardware installation and application software. The two-year courses must play an important role in the retraining of industry personnel. Thus, an extremely close relation must exist between industry and two-year institutions.

Another significant aspect of the two-year programs is that many institutions have an open admission policy. Non-traditional students are likely to be participants. Students are strongly career oriented. Many are seeking a viable migration path in their careers or a complete change of career orientation. Students must feel confident with their progress within the curriculum. Other important characteristics of two-year graduates included (1) entry level competence, (2) appropriate attitudes for success including creativity and willingness to change, and (3) awareness of the responsibilities of management.

Career Opportunities

Careers for 2 year graduates included end-user support specialists and application programmers. Other career paths include computer systems support and operations support activities. In smaller organizations, the two-year graduate may be either the database manager for commercial software, or may serve as the data processing manager.

Other Areas of Study

To support careers in Information Systems, previous authors state that problem solving and communication skills are of utmost importance (Longenecker and Feinstein, 1991a,b; Mackowiak 1992; and Hoffmann 1991). The same skills were found to be important for two-year programs, even placing them ahead of information technology (IT). Also of significance are a desired knowledge of individual and group behavior including ethics and legal issues. Accounting and management skills are expected as well.

COBOL and C are significant programming languages for 2 year students. Also of interest is the importance of the personal computing environment for both as a hardware platform and for commercial software. Other important environments include UNIX with DOS/VSE and MVS also represented.

Knowledge Clusters in the Curriculum

During preliminary meetings of the DPMA two-year curriculum committee, it was established that the knowledge clusters developed in the four-year model IS'90 (Longenecker and Feinstein 1991a,d) would, with some modification, be suitable for the two-year associate degree in Information Systems. The committee used as its starting point the IS'90 clusters. Revisions to the clusters' goals and objectives were

thus prepared and included within the survey. Figure 1 represents the clusters planned for use within the two-year committee for describing appropriate aggregates of knowledge, with approximate sequencing. (See Longenecker and Feinstein 1991a,c,d for presentation of Cluster concepts).

In its preliminary work, the committee synthesized the cluster definitions, goals and topics from IS'90. The survey was used to validate this work with the results.

Cluster A - Fundamentals of Computer and Information Systems

Cluster A defines the role of Computer Information Systems. Components of Computer Information Systems are presented. A computer system with current end-user software is used to solve problems within an organizational environment: In this cluster, the beginning student will be exposed to the historical, current and projected roles of Information Systems as they affect organizations and society. Students will be introduced to systems theory, computing systems components and systems development. Students will be introduced to the concepts of formal problems and their solutions using computer applications. The relevance of solving organizational problems will be introduced; the student will practice problem solving with computer applications.

Cluster B - Information System Concepts

In Cluster B organizational information needs are identified and the role of IS in information processing management using current and emerging methodologies is presented.

Students with previous knowledge of CIS fundamentals or equivalent experience will be exposed to the role, value, types and applications of information systems in organizations. Systems theory, systems controls and total quality management, a discussion of work flows, and the modeling of organizational data and relationships will be presented. Project management and group dynamics are introduced. System development methodology and CASE tools are defined. Computer and human interactions, security, and privacy issues will be supported through a laboratory practicum. Using and controlling Information Systems will be practiced.

This cluster provides for practical hands-on type experiences with pragmatic aspects of Information Systems within organizations. Those interested in gaining organizationally related end-user experiences would elect course(s) that implement this cluster.

Cluster C - Computer Concepts

In this cluster the function and architecture of computer systems hardware and software technologies are presented.

Students with previous knowledge of CIS fundamentals or equivalent experience will be exposed to components of a computer system. This includes computer hardware, related system software, interoperability, and human interaction. Discussion and laboratory experiences will include specifying and selecting, configuring, installing, and using computer systems to solve organizational problems.

The material in this cluster can be presented in a sequence of courses that will enable end-users to gain important hands-on experiences with computer systems hardware and systems software. This would include practical experiences with both network and multi-user minicomputer (open systems software, particularly). Data communications would be an important component of the cluster.

Hands-on experience would include actual installation of current hardware and systems software in a controlled, structured laboratory environment.

Cluster D - Applications Development / Software Engineering

In Cluster D, Information Systems techniques are used to solve organizational problems of limited complexity.

Students with previous knowledge of CIS Fundamentals, IS and Computer Concepts or equivalent experience will work with logical and physical representation, manipulation, validation, security and control of data.

Students will use project management techniques within SDLC or other methodologies to develop applications/systems of limited scope. This includes design and implementation of a database involving applications.

Applications will be developed in procedural (i.e. COBOL, C, RPG) and higher level languages (eg CASE, SQL, 4th/5th GL's) determined by the application.

Students will demonstrate the appropriate use of human computer interfaces (i.e. Common User Access {CUA}, change management, training, procedure conversion). They will propose, present and defend a computer application/system solution.

Applications involving AI/Expert systems will be discussed and demonstrated or used.

Cluster E - Systems Development/Systems Project

In this cluster Information Systems technologies are used to solve departmental-wide managerial and organizational problems. The use of project management techniques in the solution of real world IS problems of a more complex nature will be applied. An assessment of the students' level of achievement will be conducted.

Students who have developed applications/systems of limited scope will explore areas of Information Systems management within an organization (i.e. planning and control of the IS function, career planning and professional development). Laboratory experiences simulate an atmosphere typical of an entry level work environment. Experiences include group dynamics, team work, analysis of user requirements, IS planning, and feasibility. Project and time management techniques will be applied.

Summary

Five hundred two-year programs in IS were surveyed to form the basis for a model curriculum for a two-year program in Information Systems. It was considered important to have input from as wide a body as possible to validate the concepts. A response rate of 24% yielded 120 returned surveys.

The respondents indicated a desire to have the curriculum document address several items including the production of competent graduates, specific courses, and their sequencing. Any curriculum model should be able to accommodate change.

The model should provide access to a terminal degree, a path into four-year programs and an opportunity for individuals to return to upgrade their skills. A close tie must be maintained with industry.

In addition to the obvious IT skills, it was found to be essential for the students to develop problem solving skills and to be able to communicate effectively.

End-user skills were considered important. COBOL, C and C++ were considered to be the most important 3rd generation languages. Knowledge of personal computing was very important. Knowledge of UNIX and several mainframe environments were also considered important.

The survey may be used to validate the preliminary knowledge clusters and topics suggested by the Two-year Curriculum Committee.

Acknowledgements

The authors wish to thank the DPMA for its help in helping to for sponsoring this work. In addition they wish to acknowledge the contributions of the DPMA Two-Year Curriculum Committee in helping formulate the survey in and constructing the detailed cluster descriptions.

Members of the Committee include:

- **Robert Fournier:** Alpena Community College, **Chair**

- William Bonney: Hudson Valley Community College
- Daniel Claborn: Oklahoma State University
- David L. Feinstein: University of South Alabama
- Mary Garett: Lansing Community College
- Mary Jo Haught: MJ Systems
- Joyce Currie Little: Towson State University
- Herbert E. Longenecker, Jr.: University of South Alabama
- Tony Mann: Sinclair Community College
- Donald Medley: Ventura Community College
- Leon Price: University of Oklahoma
- William R. Reaugh: Caterpillar Incorporated
- Sharon Szabo: Schoolcraft College
- Julian Wade: DeKalb Technical Institute

References

[ACM 1968] ACM Committee on Computer On Computer Science. Board. "ACMRecommended Curricula for Computer Science and Information Processing Programs in Colleges and Universities", ACM, New York, New York, 1981.

[ACM 1983] ACM Committee on Computer Curricula of ACM Education Board. "ACM Recommendations for Information Systems, Volume II", ACM, New York, New York, 1983.

[Ashenhurst 1972] R. L. Ashenhurst (Ed.), "Curriculum Recommendations forGraduate Profesional Programs in Information Systems", ACM, 1972, (see [ACM 1983]).

[Couger 1973] J. Couger (Ed) "Curriculum Recommendations for Undergraduate Programs in Information Systems", CACM volume 16, number 12, December 1973, pp 727-749.

[DPMA 1981] "DPMA Model Curriculum, 1981". Published by DPMA, Chicago, 1981.

[DPMA 1986] "DPMA Model Curriculum, 1986". Published by DPMA, Chicago, 1986.

[Hoffman 1991] Gerald M. Hoffman. "What Industry wants from the Universities",Working Paper 91/13, Northwestern University, August 1991.

[Longenecker and Feinstein 1991a] Herbert E. Longenecker, Jr., and David L. Feinstein. "Draft" IS'90 A Model Curriculum for Undergraduate Programs in Information Systems, including A Survey of Undergraduate Programs of Information Systems in the US and Canada, Published by DPMA, Chicago, 1991.

[Longenecker and Feinstein 1991b] Herbert E. Longenecker, Jr., and David L.Feinstein. "A Survey of Undergraduate Programs of Information Systems in the US and Canada" Journal of Information Systems Education Volume 3, #1, Spring 1991, pp 8-13.

[Longenecker and Feinstein 1991c] Herbert E. Longenecker, Jr., and David L.Feinstein. "On establishing Excellence in IS" Journal of Information Systems Education Volume 3, #1, Spring 1991.

[Longenecker and Feinstein 1991d] Herbert E. Longenecker, Jr., and David L. Feinstein. IS'90 A Model Curriculum for Undergraduate Programs in Information Systems, Published by DPMA, Chicago, 1991.

[Mackowiak 1992] Kate Mackowiak, "Skills Required and Jobs Available for CIS Majors", Interface, Volume 13, Issue 4, Winter 1991,1992.

[Nunamaker 1982] Jay F. Nunamaker, J. D. Couger, and Gordon B. Davis. "Information System Curriculum Recommendations for the 80s, Undergraduate and Graduate programs", CACM Volume 25, Number 11, November 1982.

MAINTAINING CURRICULAR COMMONALITY AND INSTRUCTIONAL QUALITY IN A MULTICAMPUS "2+2" EDUCATION PROGRAM

Mary Martin Koleski, Ph.D.
Associate Professor
Coordinator
Computer Technology (Computer Information Systems Technology)
Purdue University
Kokomo, Indiana

ABSTRACT

"Two-plus-two" educational programs must serve on-going students who plan to complete a four year degree, as well as associate degree students who expect to seek gainful employment after graduation. Problems are added to this duality when the lower division programs are geographically remote from the campus offering the upper division courses. Control of such programs may be centralized or decentralized. Decentralized programs may be autonomous, with little or no curricular commonality. Centralized programs may strive toward more curricular commonality and instructional quality through one of two orientations: a sharing of responsibility among all faculty members or an imposition of control by the upper division faculty. This paper is based on research in one large institution of higher learning as well as interviews in and observations of two other institutions with geographically dispersed programs.

INTRODUCTION

Two-plus-two degree programs have two unique features. The first two years of study must serve as preliminaries to upper division courses for some students, while for others the lower division courses lead to a terminal Associate degree and employment. These features are complicated if the lower division courses are taught at multiple sites and the upper division courses are, for the most part, taught at one main campus. Problems arise not only over course content for graduating versus on-going students but over the maintenance of curricular commonality and instructional quality for students 50 to 150 miles away from the four-year main campus.

This paper is based on research, interviews, observations, and a study of documents over two decades in three major institutions of higher learning. Geographically dispersed programs are characteristic of each of these institutions. The solutions chosen for the problems illustrate two orientations along a centralization /decentralization continuum.

CENTRALIZED CONTROL

In a centralized pattern of control, the department on the main campus of the school has responsibility for the curriculum and decides what topics will be included, in what order and to what degree of emphasis.

DECENTRALIZED CONTROL

In a decentralized pattern of control the various campuses are totally or partially autonomous, and each campus faculty determines its own curriculum for each degree program. The individual campuses become noted not only for specific curricular content, but also for the quality of the instruction on that particular campus. The result is that employers do not ask, "What school did you attend?" but "What campus of that school did you attend?" If any inter-campus communication exists, the faculties merely "agree to disagree" about what is important for their students to learn.

ONE STATE'S "2+2" SOLUTION OF THE CENTRALIZATION/DECENTRALIZATION DILEMNA

Approximately ten years ago, business leaders in several communities around the state of Indiana requested that Purdue University bring existing technology programs to local communities. The ideal plan was that every citizen in the state eventually would be within thirty-five miles of a university-based technology program. Rather than establish thirteen new university sites around the state, the State Legislature approved and funded the Statewide Technology Program and placed curricular and administrative control for the program under the School of Technology of Purdue University in West Lafayette, Indiana. The programs were housed in existing state or private educational institutions which are responsible for teaching all the general education components of each degree program. The technological curriculum is taught by Purdue faculty. Appropriate laboratory facilities were established for the programs offered at each site. Thirteen Statewide Technology sites now exist throughout the state. Five of the thirteen Statewide Technology sites include Computer Technology (Computer Information Systems Technology) programs.

The Statewide Technology Program was established as a centralized model so that graduates from all locations would be assured of the same curricular content despite the geographic location of their class work. In addition, those who wish to complete their Bachelor's degrees on the main campus after finishing Associate degree programs at a statewide site can continue on to their upper division courses on a par with those students who completed their first two years at the main campus.

The profile of Statewide Technology students who attend the branch campuses is very different from the traditional residential campus student. While some traditional aged students (18 to 22 year-olds) attend local campuses for economic reasons, the average student age of those attending the branch campuses ranges from 28 to 33. Seventy percent of these non-traditional students are working, with over half holding full-time jobs. Seventy percent attend school part-time, at commuter campuses, and usually after 5 PM. Sixty-five percent are married and approximately 55% have children living at home. These students do not see themselves as being "pirated" away from the main campus. Some of these students

already have bachelor's and even graduate degrees and are seeking more saleable skills. Many are encouraged or even required to upgrade their skills by their employers. However, the vast majority are non-traditional students who might not attend college at all without the availability of these technology programs. While almost two-thirds of the main campus graduates leave the state to find employment, approximately ninety percent of branch campus graduates take jobs within the state, usually within 50 miles of their campus.

This paper are not based solely on the Purdue Statewide Technology Program. As was stated earlier, three institutions of higher learning were studied and the observations below are based on a combination of the three schools.

TWO ORIENTATIONS: SHARING AND IMPOSITION

Although main campus/outreach campus "2+2" models being described call for a centralized plan, some departments within schools interpret the degree of command in different terms, namely imposition of control versus sharing of responsibility.

IMPOSITION - Centralized control at its worst creates an imposition of all aspects of the program. The main campus faculty view themselves as somehow smarter, more experienced, more professional and more enlightened about what the students need than those who have contact with them daily. They were, after all, hired to teach on the main campus! The capabilities and special technical skills for which outreach faculty were hired are under-utilized. The contributions of outreach faculty members, many of whom are early retirees from industry, are denigrated or ignored. The differences in populations, hardware, laboratory hours, local industry needs, and personal or professional obligations of the students frequently are not understood and are therefore discounted. The faculty from the main campus dictates not only topics and sequence, but textbooks and even what assignments and tests are to be given to the students and when, with no input from the local faculty.

There is sometimes a lack of acceptance of the outreach programs on the part of main campus faculty. Some faculty members view these programs as an infringement on main campus enrollments, as inferior to main campus programs and as a burden on overworked faculty members to help maintain commonality.

The morale of the outreach faculty members deteriorates rapidly under an imposed centralized system. They feel that the main campus faculty members demonstrate a "We know what's good for you" attitude.

Sharing - In a department with a sharing attitude, the main campus faculty are encouraged to:

- Take into consideration the differences in student bodies as they affect assignment expectations (in quantity and length but not quality).
- Respect the capabilities of the outreach faculty since ALL were hired under identical criteria.
- Accept input from outreach faculty on common areas of course content while allowing flexibility to meet local industry needs.

- Recognize as few as 10% of the outreach students may go on to the main campus to complete their bachelor's degrees. Therefore, 90% are terminal students whose needs may differ significantly from lower division students in a four year program in terms of employment preparation.

- Recognize that on small campuses faculty must be able to teach many courses concurrently rather than specializing in one area or even one course. Their classes may have fewer students, so that the volume of grading is less, but they are responsible for three or four different course preparations each semester.

- Involve all departmental faculty in autonomous work teams and quality circles to discuss common concerns.

- Communicate!
  Communicate!
  Communicate!

ADMINISTRATIVE COMMITTMENT

The primary requirement for making any program work is to have complete and unstinting approval of the organization's administrators. If they do not believe the distant programs can be made comparable and maintained with the same quality as those on the main campus, they will not commit funds for necessary state-of-the art equipment, hire equally qualified faculty or encourage their subordinates to support their outreach faculty colleagues.

WAYS TO MAINTAIN INSTRUCTIONAL QUALITY

There are a number of means that can be encouraged to maintain curricular commonality in a geographically separated 2+2 program of studies. The following are a few examples:

- Involve both main campus and outreach site personnel in the Search and Screen process for faculty. The criteria for all hiring must be the same. An important question to ask is, "Would we accept this person to teach the lower division courses on the main campus?" (It must be remembered that the statewide faculty may need to be more generalists than specialists.)

- Maintain contacts with outreach faculty through social, cultural and personal development events. Casual conversations at informal gatherings can give insight into common problems and inovative solutions. These events must be scheduled with care since main campus classes tend to be offered in the daytime, while most classes on commuter campus are offered in the evenings.

- Use comparable criteria for student evaluations of all faculty.

- Use comparable criteria for promotion and tenure decisions about all faculty, remembering that outreach faculty may not have access to equivalent library and laboratory facilities as do main campus faculty in order to engage in publishable research.

WAYS TO MAINTAIN CURRICULAR COMMONALITY

A number of methods can be used to help maintain curricular commonality. It must be remembered that since the lower division courses are intended to feed into the upper level content, there must be consultation and agreement among all

instructors to decide upon what must be covered at the lower level to maintain a smooth transition for on-going students. The upper division faculty must, to some extent, have final say about their expectations of students entering their courses.

Some other helpful techniques are:

- Have outreach faculty serve as members of the college or university Curriculum Committees. If travel time between the campuses makes this difficult, conference telephone calls can be used during meetings to allow long distance participation.

- Have textbook decisions made by committees which have both main campus and outreach faculty participation. It may be necessary to approve more than one "acceptable" text for each course to accommodate differences in hardware, software and community employment needs.

- The department's administrators must strive to furnish comparable hardware and software configurations for all sites to simplify equal curricular demands.
- Offer seminars for all faculty who will be involved in teaching prototyped courses before they are introduced into the curriculum .

- Arrange for some new or experimental courses to be developed and prototyped by outreach faculty.

- Share instructional materials, not as mandates but as suggestions for problem solutions. Copies of all duplicated class notes and handouts can be shared by faculty at all sites.

- Share copies of quizzes and tests, again, not as requirements for usage but as suggestions. This allows all faculty to see what others emphasize and deem important.

CONCLUSIONS

"Two-plus-two" educational programs offer unique problems. Centralized versus decentralized control of the programs must be addressed. Maintaining curricular consistency and comparable instructional quality when the first two years of these programs are offered at multiple sites, some far removed from the upper division campus, presents additional problems but also offers challenges which can enrich the programs for all campuses and for all faculty involved.

BIBLIOGRAPHY

Gentry, Don K., Dean, School of Technology, Purdue University, West Lafayette, IN. Interview, June 1990.

Gross, Edward and Etzioni, Amitai. Organizations in Society. Englewood Cliffs, NJ: Prentice-Hall, 1985.

Koleski, Mary Martin. Programmatic Inconsistencies in a Multi-Campus Institution of Higher Learning. University Microfilms International, 1984.

Robbins, Stephen J. Organizational Behavior. 5th ed. Englewood Cliffs, NJ: Prentice-Hall, 1991.

THE EMERGENCE OF CASE TECHNOLOGY

Herman P. Hoplin
School of Management
Syracuse University

ABSTRACT

There is a trend toward the use of Computer-Aided Software Engineering (CASE) in systems development; however, the growth is somewhat slower than had been anticipated. CASE has become more and more vital to both large and small projects. CASE is now a technology for all software developers. Industry encourages colleges and universities to develop software engineering programs based upon automated tools in instructional systems development. Particular focus is on increasing the productivity of current and future software engineers. CASE technology is seen as one of the ways in which this can be accomplished. Now that the management of computerized information systems (IS) has become an integral function within the business establishment, IS also is undergoing the same far-reaching transformations caused by computer automation. CASE tools are being designed with the business analyst in mind. CASE tools not only help the user analyze system requirements and design the program modules, but automate other functions inherent in the IS functions. This paper focuses on these rapidly emerging developments. The author analyzes CASE as it currently stands; then looks at the benefits and drawbacks of CASE technology. Also addressed is what users are looking for in CASE and how they evaluate their current CASE applications. Finally, the author looks at the future of CASE, focusing on areas with high potential for CASE usage.

INTRODUCTION

More and more, CASE tools are being designed with the business analyst in mind, not the programmer. Industry experts are telling us that a 30% improvement in productivity can be achieved if CASE is implemented properly. (Yourdan, 1992) The ideal CASE toolset does not exist today. Considerable gaps in the sequence of CASE design remain and the necessary technology or industry-wide standard to fill these gaps is lacking. CASE tool vendors are well aware of this and are working to develop more sophisticated tools to fully integrate CASE.

At present, the CASE toolsets are strongest in the requirements-analysis and the module-design phases, the so-called front-end tools. Recently, reverse-engineering tools for the maintenance phases have started to become a reality. The biggest voids still show up in the code-generation phase and intertool communication capabilities.

CASE toolsets provide a means for fully and accurately specifying what a software system must do. The system requirements phase is the key to success or failure. At this stage, it is feasible and relatively inexpensive to change the design to satisfy new customer requirements and to ensure that the logic of the design is correct and allows for future changes.

CASE, which was once the software developer's impossible dream, is not only feasible but is now a reality. CASE tools are currently available for a wide variety of computer systems, including inexpensive minicomputers such as IBM PC-clones. To facilitate different kinds of software design, the tools use different design methodologies. As CASE tools mature, vendors are recognizing the need to develop interlocking tools to cover design and extend further into the life cycle.

The available CASE tools fall into two basic categories--first, tools that clearly define what a proposed software system will do; second, tools that specify the actual software modules for fulfilling these requirements. These provide for consistency among different sections of a system regardless of how many people are developing the system. CASE vendors use different methods on how to implement this.

Many of today's CASE tools offer you a choice among several design methodologies for two main reasons. First, such flexibility means that you will not necessarily have to learn a new methodology to be able to use a particular tool. Second, if you are already familiar with several methodologies and their notations, you can apply the methodology that best suits your problem.

CONCEPTS AND MISCONCEPTIONS OF CASE

Underlying CASE is the concept of software engineering which in turn is based on generally accepted engineering principles and practice. These principles emphasize formalism, standardized design, planning, and control in the developmental process.

It is critical to recognize that these techniques are performed within the context of a controlled, managed environment that promotes productivity, quality, and efficiency both in the short and long run. Integration of the related elements of management, planning, and development is also instrumental in software engineering. Problems are dealt within the bounds of a rational holistic approach.

Although the acronym is now widely accepted, CASE does not yet have a simple industry-wide definition. CASE tools have been defined as providing a means of standardization, discipline, and automation to systems analysis and application development. CASE is a combination of procedures, methods, and software tools that treat the entire application development process as a system that can be automated.

Generally speaking, software engineering is an enhanced means by which an organization can choose an appropriate methodology for developing information systems spanning the range of the software life cycle from the strategic planning to the enhancement of existing applications. CASE is the technology that allows for automation of certain or potentially all endeavors that are undertaken within this framework.

The application of CASE to software engineering is directed at achieving several main objectives. Namely, increasing the productivity of all individuals involved in the engineering process including systems analysts, programmer, and MIS managers. It enhances the quality of the final product and intermediate deliverables in the development process. Moreover, CASE provides for more effective management control over the development process by providing for a computerized mechanism for one or all of the activities commonly associated with software development.

CASE tools that focus on the earlier stages of system development (e.g., planning, cost appraisal, structured design) are referred to as front-end products. Those tools that assist in the later stages of development (e.g., code generation, report writing) are known as back-end. CASE products that offer an integration of CASE tools affecting all stages of the life cycle have been named Integrated Project Support Environments (IPSEs) or, alternatively, Integrated Computer Software Engineering (ICASE). Regardless of which type of CASE a company chooses, emphasis must be placed on the comprehensive approach to systems development and maintenance that software engineering theory and practice requires. As CASE tools develop and mature, these tools will extend beyond design and penetrate into the life cycle.

A CASE tool can also be classified in accordance to the general type of activity that it supports.. There are three basic activities that CASE tools can be used for: (1) strategic management and planning, (2) systems analysis/design and (3) programming and maintenance. Michael Gibson, Charles Snyder and R. Kelly Rainer have designated CASE support of these three activity groupings as 'Upper CASE', 'Middle CASE', and 'Lower CASE', respectively. (Gibson, et al, May 1989)

As might be expected given the high level of confusion surrounding CASE technology, there are many misconceptions concerning what it is and what it can do. One of the more notable misconceptions is that CASE is a replacement of fourth generation languages. As noted above, CASE is available for use in areas other than programming.

Whereas CASE provides the greater ability to track project progress and to develop strategies and plans, it does not replace the vital function of leadership and direction on the part of management. A structured environment is still necessary. CASE can enhance effective management control, but does not eliminate the need for it.

There is a tendency to believe that CASE eliminates the systems analyst. What CASE does is to give the analyst a set of tools from which to analyze and design systems, eliminate cumbersome paperwork, and makes designs more malleable in responding more efficiently to unforeseen changes in design.

The perception that the backlog of DP requests can once and for all be ended by employing CASE tools is erroneous. Even though CASE has the ability to deliver reliable systems and satisfy maintenance requests more efficiently, the sheer volume of increasing requests will not eliminate existing backlogs.

It should be recognized that CASE will not result in an immediate increase in productivity. There is a significant amount of overhead involved in starting up a CASE system. Documentation and specifications must be manually entered into a newly introduced CASE system. Productivity will generally be increased over the long run; at that point the benefits may increase geometrically if CASE is employed properly.

ANALYSIS AND EXPECTATIONS OF CASE

Simply stated, CASE technology is the collection of tools designed to assist the system professional in his duties. These duties include all phases of the systems life cycle. CASE tools can have a wide variety of functions, from simple data-flow diagram plotters to

complex system code generations.

In this analysis, users' expectations of CASE systems highlighting what they feel are the most important factors in a CASE system will be addressed as well as a look at the current benefits and drawbacks of current CASE technology. The analysis will be concluded with an evaluation of how CASE technology meets up with users' needs and expectations.

A recent survey of 800 computer installations lists the eight most important CASE tool features (ranked by respondents use). (McClure and Ambrosio, 9, 3, 1989, 33)

Checks Logic	96.3%
Supports Full Life Cycle	77.5%
Fully Integrated	74.0%
Prototyping Capabilities	72.7%
Connectivity	71.0%
Contains a Repository	68.5%
Support/Training	67.3%
Uses Standard Technique/Methodology	66.4%

Several conclusions can be reached from this data. First, users are concerned with the development of viable systems. Second, users desire tools that deal with all aspects of the systems life cycle or work with other tools to create a total package. Finally, users are concerned with the functioning of the tools, desiring packages that utilize a methodology, have prototyping ability, and are well supported by the developer or vendor.

BENEFITS

One of the primary benefits of CASE technology is that it brings about the use of a standard methodology to all systems projects. This kind of forced methodology results in a consistent product.

There are several choices of methodologies that are currently available. One of the favorites is the traditional process-oriented methodology. Many systems professionals are familiar with and have had experience using it. In addition, there are two new methodologies that are gaining in popularity for many applications: Information Engineering and Prototyping. Information Engineering is an example of a data-oriented approach to systems development. In this methodology, data design takes precedence over procedure design. First, logical models of the data used by the organization are developed; then, individual application systems are developed. With these methods, applications can be better integrated and data sharing better controlled.

Prototyping is the development of a workable system that can be used to test the functioning of the desired system. This methodology is an iterative process with the prototype being adjusted and refined in response to the user's reactions until it is found to be adequate to serve as the actual system.

Another major benefit of CASE technology is the increase in the quality of code developed for a system. (Bouldin, August 1989, 30-39) One of the primary reasons for this is the automatic error checking features that are present in most code-generating tools. Because CASE generated code is of a higher quality than traditionally generated code, it is easier to maintain. (McClure and Ambrosio, June 1989, 41)

Another benefit of CASE tools is the development code that will be reusable; i.e., code that can remain fundamentally unchanged each time the system is modified or restructured. Code generated from a CASE system still has a greater potential for reusability than traditionally generated code because of its higher quality and easier maintainability so this can still be listed as a benefit.

DRAWBACKS

The major drawback of CASE technology is the lack of industry standards for the interfacing of various CASE tools. Clearly this is an issue that must be resolved in order to make CASE a truly useful technology. One notable exception to the existing situation is KnowledgeWare's "Application Development Workbench (ADW)" which is a comprehensive collection of CASE tools integrated through an Artificial Intelligence (AI)-based repository. (Application Development Workbench, 1991, 2)

Another drawback of CASE is that there is a significant learning curve inherent in the technology. Typically, there is a two-project learning curve where productivity is very low. Only after these first projects does CASE technology begin to approach the productivity of the previously employed technique. (McClure and Ambrosio, June 1989, 41)

A third drawback of CASE technology is the cost involved with a CASE implementation. To start with, there is the initial cost of the CASE software and the hardware to run it on. In addition, there are the training costs associated with the CASE technology which have been estimated as being about three times as much as the cost of the software. (McClure and Ambrosio, June 1989, 43) Because of this significant capital requirement, CASE tools are frequently avoided by organizations that look only to immediate

benefits, rather than long-term returns.

EVALUATION

In an evaluation of CASE tool performance, one must first look at how users tend to benchmark CASE effectiveness. In a recent survey, experienced CASE users ranked three decision criteria with the following results: (Bouldin, August 1989, 31)

Primary Criteria:		Secondary Criteria:	
Time Savings	38.8%	Time Savings	47.5%
Qual. Improve.	39.2%	Qual. Improve.	36.9%
Financial Savings	13.3%	Financial Sav.	24.6%

Tertiary Criteria:

Time Savings	13.8%
Quality Improvement	21.4%
Financial Savings	61.9%

When viewed in this light, CASE technology looks pretty good. CASE technology offers significant time savings once the learning curve has been overcome and it has a proven record in the area of code quality improvement.

Another way to evaluate CASE technology is to look at it in terms of the eight important features that were previously discussed. In this manner one can evaluate how CASE is meeting users' needs rather than how users evaluate CASE--on four of these major features, CASE is currently performing very well; on three of these major features, CASE is currently not performing up to the users' expectations; on one, the utilization of a repository, does not fall into this evaluation unless it is important to the individual user.

As was discussed in the drawbacks section, although many CASE tools support the full development life cycle, the performance of individual elements of these tools is usually not up to the users' standards. Also, with the exception of vendor development such as KnowledgeWare's ADW cited previously, CASE tools are poorly integrated and vendors currently offer little connectivity with other vendors' tools.

To conclude this evaluation, it appears that CASE tools generally are meeting the users' expectations and requirements fairly well, but there still is room for significant improvement. A Buyers' Scorecard poll shows that overall user satisfaction with leading I-CASE tools is up from an averazge score of 69 in 1991 to 72 in 1992 against a total possible score of 100. (Slater, August 1992, 81) CASE tools are now strong in many of the technical aspects of the systems development process. However, they are still very weak in one major area of importance to the user: integration and connectivity. In order for CASE tools to be truly viable in the future, it is very important for this area to be addressed.

FUTURE OF CASE

As has just been discussed, CASE, although an impressive technology, still has some problems to be worked out in order to make it truly invaluable to the systems professional. These problems are not insurmountable, but they do represent some of the major barriers against universal CASE adoption. However, if these problems are solved, many areas will be of major consideration to the CASE technologist.

Three of these areas, Prototyping, Expert Systems, and Project Management, that represent some new approaches of CASE technology are discussed below:

Prototyping. Prototyping is the process of developing a simplified early version of a system to use in building desired improvements. Although prototyping is a currently utilized methodology in CASE technology, it is an important area of massive potential for the future, that involves Software Evolution and hybrid technologies.

Software evolution refers to all activities that alter a software system, including requirements changes, performance changes, and repair. Essentially, it is systems maintenance in its broadest scope.

Object-oriented CAD is truly a hybrid technology as it represents the application of CASE technology to the field of circuit design. It will be of increasing interest in the future as it allows for a convergence of knowledge between software and hardware developers.

Software storming is another hybrid technology as it represents the combination of knowledge engineering (the brainstorming problem solving technique) and systems development technology.

Expert Systems. Expert systems are another area that has potential to reap enormous benefits from CASE technology. Traditionally, many of the most successful expert systems were developed by a trial and error methodology. It is felt that taking a more structured approach in the development of expert systems will allow the designers to avoid the mistakes of the past. It is hoped that CASE technology will become more involved with this structure.

Project Management. One of the biggest problem areas in MIS is project management. There are horror stories of numerous incidents of runaway development projects which have adversely affected budgets, reputations, and even competitive capabilities of the companies involved. (Levine, March 1989, 32) The integration of project management techniques into CASE methodologies would help control many of these projects.

It is becoming increasingly clear that management wants software developers to have the same sort of accountability that other functional areas have for their projects.

This is an area that can be addressed by CASE technology. CASE, because of its methodological basis, already contains much of the structural design of a project management system. It is just a matter of merging these two areas to make a simple consolidated product that can handle total life cycle design and project management. Currently there are many systems that purport to offer CASE project management (Levine, March 1989, 35-38) However, because of the current lack of integration between CASE tools, most of these packages cannot satisfy the requirements of both the developers and management. It is for this reason that project management is considered an area for future development in CASE. It is hoped that once the integration problems are solved, we will be able to develop truly useful CASE tools that will satisfy almost everyone.

The development of more end-user oriented CASE tools is also in order. A similar development occurred when spreadsheet programs brought significant computing power to the hands of the end user by simplifying the process of generating financial and other number-intensive applications. This, development could be considered a good possibility as CASE tools tend to be very complex instruments, similar to many accounting programs that were very complex until the development of Lotus 1-2-3.

CONCLUSION

Since the arrival of improved microcomputers in the mid-1980's, CASE tools have become the dominant trend in systems development.

User expectations of CASE tools are very high and are currently being only partially met. Benefits include standard methodology, increase in code quality, and the development of reusable code. Drawbacks of CASE include lack of integration of tools, a high learning curve, and high costs.

CASE technology is facing many opportunities for further development in the future. The decade of the '90's can be viewed as the beginning of the third generation of CASE. There are many areas that can be further developed, including Prototyping, Expert Systems, Project Management, and Repository Products.

Organizations around the world are now using CASE. In this paper, we have established guidelines for evaluating CASE tools; what remains is further research in the development of the technology in order to influence management to make wise choices in the adoption and use of CASE tools.

In final conclusion, CASE is a technology that is truly at a critical point in its development. It has had some degree of success, but it still has much potential that it has to live up to. As CASE technology matures, it is expected that it will become an even more dominant trend in software development of new systems and a positive force in renewing the effectiveness of aging systems. Now that the CASE vision has been created, its universal acceptance requires selling it through education.

References

"Application Development Workbench," A 1991 CASE Tool Set brochure, 25 pages, KnowledgeWare, Inc., 3340 Peachtree Road, N.E., Atlanta, GA 30326.

Bouldin, B. "What Are You Measuring? Why Are You Measuring It?" Software Magazine 9, 10 (August 1989) 30-39.

Gibson, Michael L; Snyder, Charles A; Rainer, R. Kelly, Jr. "CASE: Clarifying Common Misconceptions," Journal of Systems Management (May 1989), p. 12-19.

Levine, H. "Two Separate Worlds Moving Slowly Closer," Software Magazine 9, 3 (March 1989), 32-40.

McClure, C. and Ambrosio, J. "Methodology Thumbnail Sketch--Essentials of Development," Software Magazine 9, 7 (June 1989), 33-42.

McClure, C. and Ambrosio, J. "The Most Important CASE Tool Features," Software Magazine 9, 3 (March 1989), 33.

Slater, Derek, "Buyers' Scorecard," ComputerWorld (August 3, 1992) 81.

Yourdan, Ed. "Chairman's Address", CASE World Conference, Santa Clara, CA. (February 18, 1992)

APPLYING CASE TOOLS IN DEVELOPING A MANAGEMENT STRUCTURE TOOL FOR THE UNITED STATES ARMY CORPS OF ENGINEERS' CONSTRUCTION ENGINEERING RESEARCH LABORATORY

AUTHORS

DR. DAVID C. WALLACE
ASSISTANT PROFESSOR
APPLIED COMPUTER SCIENCE DEPARTMENT
ILLINOIS STATE UNIVERSITY

LYNNE J. MIKULICH
PRINCIPAL INVESTIGATOR
UNITED STATES ARMY CORPS OF ENGINEERS
CONSTRUCTION ENGINEERING RESEARCH LABORATORY

Research Associates

LORETA HSUI
UNITED STATES ARMY CORPS OF ENGINEERS
CONSTRUCTION ENGINEERING RESEARCH LABORATORY

CARL CAMP
UNITED STATES ARMY CORPS OF ENGINEERS
CONSTRUCTION ENGINEERING RESEARCH LABORATORY

ABSTRACT:

The Applied Computer Science Department at Illinois State University is working on a research project to model an information system with the Army Corps of Engineers, Construction Engineering Research Laboratory (CERL), using a CASE tool. The focus of this research is on the Resource Management information system within CERL. A model of CERL was completed at the highest level (Level 0) but only the Resource Management information system was completely decomposed to the primitive level (lowest level). C E R L anticipates several advantages from modeling their information systems. The most important advantage is creation of a framework in which CERL can integrate all its information systems into a cohesive unit. This CASE Model can be used to evaluate the effectiveness of CERL in meeting its goals and objectives for both short- and long-term results. It can also be used to map changes in CERL's information systems as a result of new requirements and contingencies.

Possibly the one most important advantage of CASE tools for CERL (or for any organization) is the ability to implement and integrate new technology. The ability to visualize the organization in terms of its goals/objectives and the information systems that support them provides the organization with a tool that can graphically demonstrate the effects of technological changes from the highest levels to the lowest levels in the organization. Organizations can develop technology based on needs, rather than having to find a need for new technology.

INTRODUCTION

Illinois State University's Applied Computer Science (ACS) Department is working with the United States Army Corps of Engineers' Construction Engineering Research Laboratory (CERL) to develop a management structure tool using CASE products. Operating since 1969 under the direction of the Department of the Army, Office, Chief of Engineers (OCE), CERL currently supports a broad area of research. A research grant over the past year involving the ACS Department and CERL has funded the development of a management structure tool which would better utilize CERL's limited resources. This research project involved the modeling of CERL's information system structure using a CASE tool. Figure 1 illustrates the general direction of this research effort.

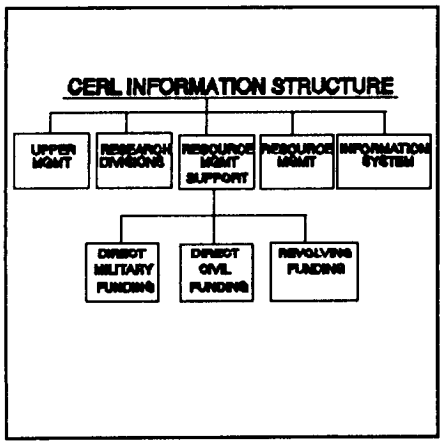


Figure 1. CERL Information Systems Structure

This research project involved several stages: (1) investigating, (2) modeling CERL's existing structure, policies, and procedures, (3) identifying new user requirements and expected benefits, (4) modeling a new information structure, and (5) generating a preliminary schedule of installation and implementation. The current research grant covered the first three stages. Subsequent research grants will cover the final two stages along with the development of the other information systems within CERL (See Figure 1).

Investigation Stage

The investigation stage involved identification of the overall goals of CERL and determination of the scope of the research project. CERL is a research arm of the United States Army. Located in central Illinois, it is specifically intended to work with educational institutions throughout the United States. Research efforts have covered a broad spectrum of current topics such as hazardous waste, waste management, facilities operations, environmental impact studies, and information system management. CERL goals are best illustrated in their mission statement.

CERL MISSION

> To provide research and development to support Army programs in facility construction, operations, and maintenance in the United States, overseas, and in Theater of Operations by achieving better quality vertical construction with attendant environmental safeguards at the least cost in time, money, and resources (CERL's Information Systems Plan(1)).

The next step in the investigation stage was to identify a proper scope for using CASE tools within CERL. Through discussions with the principal investigator and

representatives of the Information Management office of CERL, it was determined that the Resource Management Support information system would be appropriate since its functions are closely integrated with all the other information systems within CERL. One of the major goals of this research project is to help better manage CERL resources with the help of CASE tools. Thus, to demonstrate the effectiveness of CASE tools for an important information system within the organization would be helpful in getting the support from the other information systems closely integrated with this system and therefore, receiving some of its benefits. It will also help in getting upper management support which is critical if the organization is to receive the full benefits of CASE tools.

Existing System Stage

Modeling the existing structure of the Resource Management Support system involved the design of the current procedures and policies. This entailed collection of all input and output documents. Each document indicated a series of processes which involved either the generation, collection, updating, or verification of information. These documents also pointed to various agencies, departments, and other organizations which interface with the Resource Management system. Thus, after reviewing these documents and interviewing the individuals operating within the system, a preliminary model of the existing system was generated. Figure 2 represents the general organization of the model.

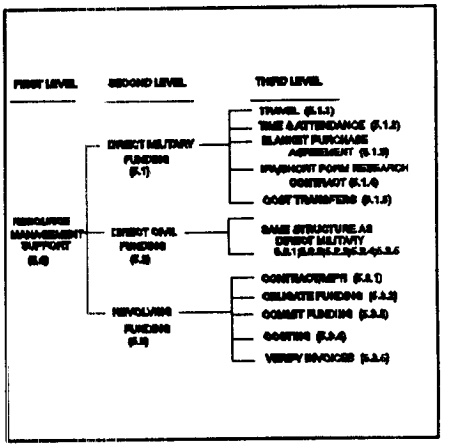


Figure 2. Resource Management Support Structure

Each process on level 2 (see Figure 2) was further specified within the data dictionary to include how data were actually manipulated within the process. For example, providing Travel Support involves the committing of funds when a research proposal has been approved and accepted by CERL and the Headquarters Department of the Army. CERL accesses a database system known as COEMIS (Corps of Engineers Management Information System) located in the disbursing office outside of CERL (Waterways Experimental Station - WES). COEMIS is a menu-driven system that allows CERL to track different types of research funding projects (Military, Reimbursable Funding, and Outgoing Reimbursable Funding). The travel funds within a research project are actually committed by accessing COEMIS and using a menu option to set aside the amount of travel funds identified on a special contract (FAD - Funding Authorization Document). When

travel orders are received, COEMIS is again accessed to obligate the amount of funds spent on the specific travel order. Thus, the obligation of travel funds is matched against the committed fund until the fund is exhausted. WES is the actual disbursing office for CERL. Therefore, WES actually pays the amount on the travel orders (this process is generally correct for all funding expenditures - Labor and Materials). WES will send periodical updates (print-outs) listing each research project and the amount of funds which are committed, obligated, and eventually paid out. WES will also send Resource Management office individual listings of each project of the committed, obligated, and paid-out funds. These listings are eventually given to each project team leader (principal investigator) as a way to help control his/her research project. This detailed process for handling travel funding for research projects was specified using a structured English format. Each primitive process was specified in a similar way.

The full model of CERL will be completed in future funding proposals. The methodology used to complete the Resource Management Information System can be used to complete the entire information system for CERL. The methodology used to complete the Resource Management Support System can be used to complete the entire organizational model for CERL.

New Requirements Stage

CERL expects several benefits from the modeling of its information systems with CASE tools. One of the most important benefits is a framework in which CERL can integrate all its information systems into a cohesive unit. The structural model (the context diagram, data flow diagrams, the data model, and the data dictionary) is a strategic management tool that will act as a blueprint for CERL. This blueprint can be used to evaluate the effectiveness of CERL in meeting its goals and objectives. It can also be used to map changes in CERL's information systems as a result of new requirements and contingencies. Thus, the structural model cannot be static; it has to be a dynamic tool to meet the challenges of changing environments.

With the aid of the computer, possible alternatives and configurations of different information systems can be constructed and evaluated without committing large sums of money to actually changing the information systems. Tests and technical walkthroughs can be used to verify the results of changes before trying to install or implement any new ideas. A more thorough front-end analysis can be completed to avoid the errors that are often overlooked during the analysis and design stage of any system design project.

The information system model can illustrate the generation and use of data needed for the day-to-day operations of an organization. These diagrams can be used by the organization to evaluate its efficiency in using information for its operations. For example, bottlenecks can be

identified within individual systems, and corrected, by examining processes where man and machine boundaries might cause an overload for the system.

The value of using CASE tools to model information systems goes beyond just the efficiency of day-to-day operations. Diagrams of the model can also model processes where reports or summaries are generated to monitor and control the operations of the organization. To insure that these systems are meeting their objectives, control mechanisms can be established to bring the system or any part of it within acceptable performance levels. For example budgets are often used to monitor fiscal performance. The generation of such budgets is an important part of the modeling process. CASE tools can be an enormous help in getting CERL's budgetary cycle under control (and making sure that the budget reflects the current needs of the Army) by insuring that only up-to-date information is used in the budget processes.

Decision support mechanisms can be modeled using CASE tools. CERL's structural model is a road map that allows for the identification of key decisions and the important factors that comprise the basis for these decisions. The structural model becomes an important tool to pinpoint strategic locations throughout the organization for decision support tools.

Another vital benefit of CASE tools for CERL (or for any organization) is the ability to implement and integrate new technology. The United States is one of the major sources of new technology in the world, yet our ability to integrate and implement this technology must be improved if we are to compete successfully in local and world markets.

The ability to visualize an organization in terms of its goals and objectives, and the information systems that support them, provides a tool that can graphically demonstrate the effects of technological changes from the highest levels to the lowest levels in the organization. These changes can be mapped on the structural model for evaluation and decision making. The introduction of new technology can shift from a crisis management situation (where companies are forced by competition from other companies to make changes) to a more planned management process. Organizations can develop technology based on needs, rather than having to find a need for new technology.

Summary

The Applied Computer Science Department at Illinois State University is working on a research project to model an information system with the Army Corps of Engineers, Construction Engineering Research Laboratory (CERL), using a CASE tool. A model of one of CERL's critical information systems was completed and will be used as a blueprint for the generation of an organizational model. This model will help CERL control and manage their limited resources more effectively and improve their decision making capability.

Practical Advice for Implementing the 1991 DPMA Model Curriculum in Small University

Eli Boyd Cohen, Ph.D., CDP, CSP, CCP, CQA
Associate Professor of Computer Information Systems
College of Business
Eastern New Mexico University
Portales, NM 88130
(505) 562-2066
email: eli@bradley.edu

Abstract

This paper describes the efforts of one small school to implement the 1991 DPMA Model Curriculum. The paper offers practical advice based on our experiences, both positive and negative. The paper describes the school and program from the viewpoint of the author, a new faculty member brought in to update the program. The paper then describes problems and tactics in dealing with the curriculum committee, scheduling, and staffing. The paper concludes with an outline of what needs to be done after curriculum revision is adopted.

The Small School Problem

Adapting the 1991 DPMA Model Curriculum requires IS departments to make several changes. While the changes required are not trivial for any university, the improvements have a significant impact on the small university, in terms of both staffing and scheduling. This paper describes those changes for one subject university.

Description of the Subject School and Program: A Biased View

The description of the school and program offered below is that of a new faculty member brought in to revise the existing program and so is biased accordingly.

The University has several campuses. The main campus, about which this paper describes, serves approximately 4,000 students. Of this number, the College of Business serves under 1,000. The University is isolated, both by location and by resources. It is located a four-hour drive from the nearest major city. In terms of resources, it is even more isolated. The faculty do not yet have access to Internet or Bitnet. Computer and library resources are quite modest. This state school plays the role of step-child, offering faculty salaries about 14% under market. The normal teaching load is four sections with two or three preparations. These situations have led many excellent faculty to move on to greener pastures.

In the past, few if any of the faculty teaching in the Computer Information Systems area have been terminally qualified to teach in the area. This has lead to problems. Students graduate from courses without developing the knowledge base and skills designated for those courses. For example, one Introduction to MIS course taught just BASIC because the economist teaching the course had limited background in CIS.

As in most small universities, we have 3-4 faculty members to teach CIS. Frequently, some are

from other business disciplines and are "borrowed" to teach a CIS class. What is more, to offer all of the courses that our students need to graduate, it has not been uncommon for faculty to voluntarily take on four preparations.

With these handicaps, it is understandable that the curriculum has gone unrevised. The old curriculum is based on the 1985 DPMA model curriculum and, with minor exceptions, shows its age.

I was hired to improve this situation. This paper describes my experiences in the Curriculum Committee and my efforts at scheduling and staffing. It also outlines the plans for next year.

Curriculum Committee

Figure 1 shows the existing (old) Computer Information Systems (CIS) curriculum for the major. In developing the proposed (new) CIS curriculum, I was guided by the following, sometimes contradictory, principles:

- Follow the DPMA model wherever it fits.
- Make the fewest possible number of changes to the existing curriculum.
- Remove administrative roadblocks, such as unnecessary prerequisites, that hamper absorbing transfer students. The old curriculum's prerequisites required the student to take CIS courses for no fewer than 2 1/2 years
- Ignore historic compromises to the curriculum that adversely affected it. Examples of this include a CIS requirement of a third economics class and handing over to computer science almost all of our elective courses.

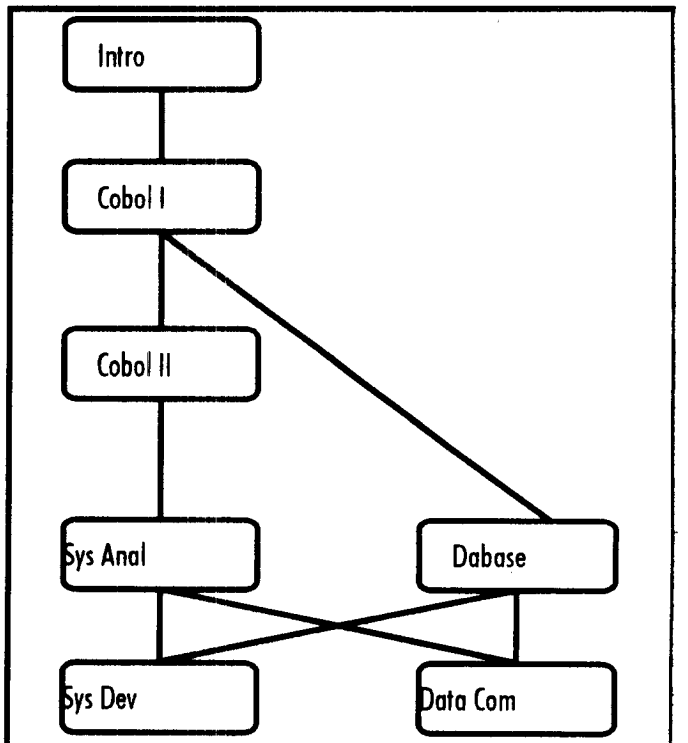


Fig. 1. The Old CIS Curriculum
(Core Courses Only)

Figure 2 shows the curriculum as adopted by the Curriculum Committee.

The new curriculum involved developing new courses, revising course titles and descriptions, and deleting some existing courses. The changes were developed in consultation with the CIS faculty and then presented to the Curriculum Committee. The Curriculum Committee then made suggestions. I would incorporate these suggestions and bring to the committee a new draft. The configuration management task became horrendous, resulting in the committee's secretary losing track of what was the final version. To make matters worse, since the process took several months, decisions and compromises that we agreed to in the early months were re-examined later in the process. If I had this to do all over again, I would have taken clear notes of the committee's actions course-by-course, and not assume that others would be taking such notes.

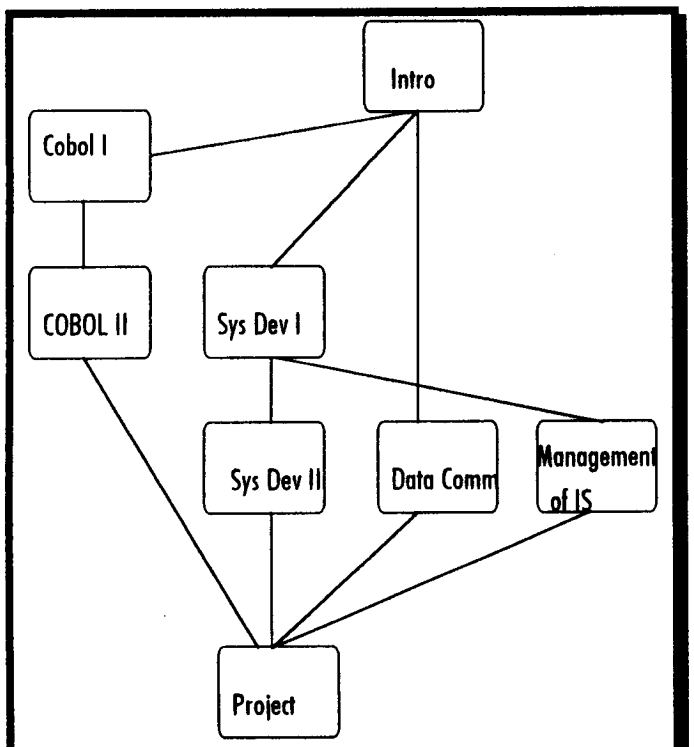


Fig. 2. The New Curriculum Following the 1991 DPMA Model (Core Courses Only)

I also learned that I needed to educate my colleagues as to what CIS is and is not. To my surprise, few of them had ever taken a class in MIS or CIS. This ignorance of the field did not necessarily preclude individuals from holding strong, albeit outdated biases, for example against BASIC.

Likewise, the curriculum suffered from some historical anomalies. The old major required a third course in economics, which I wanted to replace with a CIS elective. In the end, the committee compromised to let the student take either a third economics course or a CIS elective.

All in all, the curriculum adopted was about 90% of what was proposed. A tactic I believe contributed to the success was individual meetings with members of the committee before meetings to iron out any problems. This approach is helpful, but not a panacea. There are always those who appear to agree, but during meetings bring up new questions or objections.

Scheduling

A great problem in scheduling at a small institution is how to offer all the courses students need for graduation while also dealing with the constraints of limited numbers of faculty. An example may help to clarify this problem.

We teach two consecutive courses in COBOL, which we might call here COBOL I and COBOL II. At the present, we teach both of these two courses each semester. To relieve the strain on the single faculty member available to teach COBOL, the proposed schedule calls for teaching the first course in the Fall only and the second in Spring only. The problem is how to determine the first semester that we can implement this change in scheduling without penalizing students. We want to teach the first course in the Fall. However, since we taught the first course last Spring, its graduates need the second course in the Fall. The soonest that we can implement the Fall-Spring sequence then is a year from now, as shown in Figure 3.

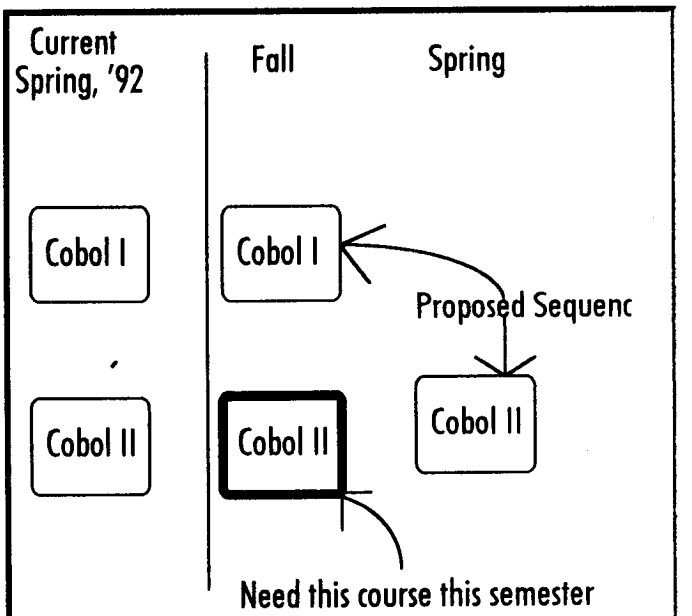


Fig. 3. Proposed Spring/Fall Sequence for COBOL meets Historical Reality

We have chosen to offer a variety of electives to remedy the complaint about our current program lacks those electives. Of necessity, we will offer these electives on a rotating basis only once every two years. This limit and the one above is due to constraints in staffing.

Staffing

My view as a newcomer is that the old maxims around here were "the student is our customer" and "give the student what s/he wants". Given our limited staff, I think that we need to revise our motto to be "let's work together to provide our customers (the students) the best program we can support".

One aspect of this is to provide incoming students with a schedule that reliably shows what courses will be taught when. In the past, the consequence for lack of planning, either on the part of the faculty or the students, was that students would need a course not offered that semester. Faculty would volunteer to teach four preparations to accommodate these students' needs. My view is that such flexibility is destructive of faculty moral and is entirely unnecessary. Overburdened faculty, no matter how dedicated, cannot give quality service to students.

As positions open, I am recruiting for generalists, for those individuals who can and will teach a variety of courses. With only three CIS positions we need to be flexible. I want the faculty in CIS to teach no more than three preparations a semester and six preparations in two years. This will enable our faculty to do some planning activities that will not only improve each course and the curriculum as a whole, but also raise moral.

A Year for Planning

We who teach in small colleges should not feel like step-sisters when it comes to developing courses that follow the DPMA model curriculum. We are better equipped to do so than those with a dozen or so MIS faculty. Here is why.

One of the core elements of the new curriculum is the integration of subject matter. For example, in place of three separate courses: Systems Analysis, Systems Development, and Database, the new curriculum offers two coordinated courses: Systems Development of Single User Systems and Systems Development of Multi-user Systems. This novel approach will make those who specialize in database or analysis a bit uneasy. It requires that the individuals teaching these courses to coordinate with other faculty at a very close level. With a small faculty this is easier, especially so when the same person teaches these courses.

Following the approach taken at Alverno College (a pioneer in innovative education), we will be developing a mission statement and objectives for our discipline. For each course those same objectives will be restated, describing how this course promotes those objectives. These objectives will drive how we evaluate the students in the course.

Unlike before, we will explicitly state that our objectives include skills in oral and written communication, group work skills, and other non-technical skills that CIS workers need, but that are not technical areas of CIS per se.

This approach will change how we teach those courses, such as COBOL, that will not be otherwise affected by the new curriculum. We will also be developing courses new to us (and some that are new to CIS in general), such as Management of the Information Systems Function and Systems Development for the Single User System. We want each of those courses to promote our mission and so they will be designed by our faculty working in concert. We want our course to work in synergy with one another. Accomplishing this will be easier at a small school. We do not have the in-fighting common to large programs. We do not have the specialists fighting for their own purposes.

Conclusion

We have seen the trials and tribulations of changing the CIS curriculum at one small university. Like most public schools today, our school is not wealthy. We have the same process of dealing with curriculum committees as do all schools. Our scheduling and staffing requirements are special because of our limited resources. This paper describes some of the approaches we have tried. Hopefully, the reader may benefit from our experiences.