

---

## PART II. Teaching Strategies and Lab Preparations

### 1. Teaching Multiple Languages in a Single Course: Our Experiences

In this part, we first introduce our experiences in teaching multiple computer languages for MIS majors in one single course. The following is two article published in *Journal of Information Systems Education* in 2002 and 2012 respectively. The first article reports an introduction programming course, and the second one repots an advanced programming course on server-side programming. These two articles document our experiences and opinions of the challenging teaching approach. Note that COBOL is no longer included in the current textbook, and is no longer taught in our program.

---

---

#### **An Approach to Teaching Multiple Computer Languages**

(Article published in *Journal of Information Systems Education*,  
Volume 12, Number 4, January 2002, pp.201-211.)

#### ABSTRACT

In the digital economy era, business information systems students need to be knowledgeable of multiple computer programming languages in order to meet the requirements of computer literacy. This paper outlines the pedagogy of an innovated course of multiple computer languages for business students in the major of information systems. It discusses the rationale of why the proposed pedagogy is different from and better than traditional ones, and describes the approach to teaching this course. Based on our experiences in the past three years, it is concluded that a single course of multiple computer languages is useful and feasible.

#### INTRODUCTION

There have been dramatic changes in information technology during the last decade. Most notable is the advancement of computer literacy of millions of information systems (IS) professionals in business and management as a result of the proliferation of electronic commerce. The changes have considerable implications for institutions of higher education whose responsibility is to train the next generation of business IS professionals. In our view, business IS students must acquire fundamental theories of IS as well as essential practical skills in computer applications. They must also develop the life-long learning ability in information technology during their business education. Recently, there is a considerable need for redesigning business IS education curriculum (IS2002 2001). Academic institutions are required to

pay increasing attention to courses of practical IS skills based on long-term strategic considerations. This paper is to report how this challenge was met by designing the contents of an innovated course of multiple computer programming languages for business IS students.

### **THE INNOVATED COURSE OF COMPUTER PROGRAMMING LANGUAGES**

The course described in this paper is entitled "Programming and Problem Solving," and is designed for sophomore students in the major of business information systems. There is no specific prerequisite other than a course of introduction to business. It is taught over one semester, normally 14 weeks. In its design, this course consists of two distinct modules. The Teaching Module provides an overview of representative computer programming languages in business computing. The Project Module involves hands-on projects.

The teaching of a comprehensive computer programming course is complicated by the extensive nature of the subject. As no report on teaching multiple computer languages in a single course has been found in the literature or on the Internet, the selection of computer languages for the course is the crucial task for the pedagogy design. The fact is that, in the modern computer age, the development of IS still heavily relies on applications of third generation computer languages regardless the advance in fourth generation computer languages and a variety of software packages. Interestingly, this is truer in Web-based applications development. To meet the challenge of the ever-changing computer technology, business IS students must be computer-literate in terms of understanding major programming languages. On the other hand, they cannot afford to learn multiple computer languages on the one-language-one-course basis. The key solution to this problem is to make a pedagogical paradigm shift and develop an innovated course.

We design the components of this course based on the skill demands from the IS job market. There have been many surveys about hard skills required for IS jobs. According to Papp's survey (Papp 1998) for the New England region, the skills in demand as found in the classified IS job advertisements are 16.0% COBOL, 13.4% C++, 10.8% Visual Basic, 9.3% C, 5.0% Java, and 3.0% HTML, which comprise the total of 57.5% general design and development skills. Taking this into account, six major computer programming languages were selected for this course. They are COBOL, C++, HTML, JavaScript, Java, and Visual Basic.

In terms of training of information technology, the goal of education for business IS students is quite different from that for computer science students, although the boundary between the two may not be clear-cut. In computer science programs, the major goal of programming courses is to teach students how to decompose a task and then map this plan into constructs of the target programming language (Sleeman 1986). Task decomposition depends on the problem domain, and there are so many problem domains are essential for computer science students, including data structure, scientific computation, file processing, Web applications, multimedia, etc. More importantly, the constructs must be available in the particular programming language. For these reasons, computer science programming courses are typically designed on the one-language-one-course basis and emphasize the syntax of the language. Also, it is common in computer science programs that a first course in programming is not influenced by a specific programming language (Malik 2000). A recent study of teaching computer science suggested to keep things as close to pure logic as possible and to treat computer languages as notational systems (Palma 2001). Apparently, the goals and process of teaching programming in computer science programs are not

fully applicable to the business IS education. We would like to teach business students to be more business oriented and business problems driven. Instead of teaching general technical details, such as formal syntax and computational algorithms, we should teach specifically how to match typical business problems and their computer solutions.

Due to time constraints, it is impossible for students to learn all these computer languages in great details. Nevertheless, students in this course are expected to have a bird's-eye view of computer programming languages as well as to develop practical skills of programming. The objective of this course is that students will understand the characteristics of traditional data file processing in legacy information systems, the philosophies of structured programming and object-oriented programming, the means of Web page development for the Internet, and the concept of human-computer interface design and decision support systems. The central methodology applied to this course is case study. Specifically, we teach typical problems of business computing and their solutions of these computer languages.

### **MOTIVATION OF THE PEDAGOGY DESIGN**

Currently, many business undergraduate programs offer courses of computer languages to IS majors. However, these courses are usually single-language courses. During the past years, we have identified opportunities to develop the course of multiple computer languages for business IS students, and have found the new design of such an innovated course have many advantages, as discussed below.

#### **The Needs for Learning Multiple Computer Languages**

Due to historical and technical reasons, there have been many computer languages in the information age, and each computer language serves its particular purpose. The needs for learning multiple computer languages can be perceived from a variety of views. From the perspective of basic job skill requirements for IS students, COBOL is still important to learn since many legacy systems are COBOL-based. However, modern Internet-based computing heavily relies on HTML, JavaScript, and Java. Furthermore, from the viewpoint of learning IS tools for management information systems and decision support systems, Visual Basic is more important than other languages. Nevertheless, from the standpoint of computer languages themselves, C and C++ are fundamental in the software industry, and are the typical sister languages of the two different programming paradigms: function-oriented and object-oriented. Given that fact that almost every textbook of information systems analysis and design includes material on object-oriented approach, IS students must learn the basic concept at the language level.

There is little doubt that knowledge of various computer languages can contribute to the learning and further career development for IS students. Then, the question is how we can teach them in a feasible way.

#### **The Feasibility of Teaching Multiple Computer Languages**

It is impossible for us to teach multiple languages by using the traditional one-language-one-course approach. To compact material of multiple languages into a single course, we must identify what concepts are essential for business IS students, and how these concepts can be delivered. After we reviewing the IS2002 model curriculum (IS2002 2002), we determined that the following concepts are essential for IS students.

- Data processing (file systems);

- Simple data types and string manipulation;
- If-then control structure;
- Loop;
- Function calling and function orientation;
- Object orientation and message sending;
- Web page development;
- Client-server computing;
- Graphical user-computer interface development; and
- Various environments for computer languages.

The relationships between these key concepts offered in individual categories of languages and the general requirements of IS education is summarized in Table 1. As shown in Table 1, the IS2002 model curriculum defines computer literacy as a set of knowledge elements related to problem solving using the computer technology. Most knowledge elements listed in Table 1, such as abstract data types, modules and coupling, software development, human-computer interfaces, object-oriented methodologies, and programming languages can be acquired only through the learning of computer languages. Hence, being computer language capable contributes to computer literacy significantly.

We consider many concepts included in traditional language textbooks, such as pointers and sophisticated data types and structures, to be less important for IS undergraduate students and not be emphasized in this course. To better teach these essential concepts of languages, we must change the traditional teaching approach that begins with syntax and ends with disjointed examples of syntax explanations and use cases. The motivation to use the case study method was triggered by the fact that our IS students were generally unsatisfied with the programming course that had been taught using the traditional approach before the redesign. Appendix A shows such a typical case of payroll processing for COBOL data processing.

Despite the variety of syntax in the different languages, languages share many common key concepts, such as if-then control, loop, and function-calling/message-sending. Once a key concept is introduced in one language, it should be understandable for students in another language.

### **The Pedagogical Shift Adds More Value to Student Learning**

This innovated course is fast-paced, and encourages self-learning on the students' part. Since the nature of this course is practice-oriented, multiple small-scale projects are required. The central point of the pedagogical shift from one-language-one-course to multiple-languages-one-course is to add more value to student learning by giving students discipline in self-learning and encouraging them to apply newly learned knowledge to the real world. According to our observations, students might feel stressful in any programming course, but never felt bored in this course. Next, we describe the course design in more detail.

Key Concepts	Requirements of IS Education (IS2002 - Competency Level and Body of Knowledge Elements)	COBOL	C/C++	HTML JavaScript Java applet	Visual Basic
--------------	---	-------	-------	-----------------------------------	--------------

Data processing	4 1.2.1 Formal problems and problem solving	X			
Simple data types	4 1.2.4. Abstract data types	X	X	X	X
If-then control	4 1.2.1 Formal problems and problem solving	X	X	X	X
Loop	4 1.2.1 Formal problems and problem solving	X	X	X	X
Function orientation	3 1.2.4.4 Modules and coupling.	X	X		
Object orientation	3 3.3.6 Object-oriented methodologies		X	X	X
Web page development	4 3.9.7 Software development			X	
Client-server computing	3 3.1.2 Systems concepts			X	
Graphical user-computer interfaces	4 3.9.6 Human-computer interfaces				X
Environments of languages	3 1.3.7 Programming languages, design, implementation.	X	X	X	X

**Table 1. Relationships between the Key Concepts and Requirements of IS Education**

### TEACHING MODULE

For a long time, it has been difficult to find a single integrated textbook that meets our needs. In order to give students a comprehensive guideline for their studies, a lecture note was developed for this course during the past years. This manuscript has been revised many times, and has recently published as a textbook (Wang and Wang 2000) that can be downloaded from the Internet.

The Teaching Module is divided into four units based on the categories of business computing problem solving, each of which is followed by a written exam. The features of the course competency, relevant paradigms, and business applications are presented below.

**Unit 1: Data Processing and COBOL**

File processing is an essential type of business data processing. COBOL-based file processing is the backbone of legacy information systems. In this unit, the techniques involved in file processing, such as three organizations (i.e., sequential, random, and indexed), are discussed. Because these techniques are built in COBOL programming, emphasis is placed on the comparison of the three file organizations in terms of their advantages and disadvantages in various circumstances of data processing.

Since COBOL is still a commonly used computer language in the business community, and is the first computer language for most business IS students to learn, about 29% class time is devoted for COBOL. Instead of emphasizing the syntax, we present four typical COBOL programs related to payroll processing. These four typical programs give students an undivided prototype of business data processing functions: creating a master file, creating a transaction file, manipulating these data files, and maintaining the data files.

The teaching emphasis is placed on the following four aspects of COBOL: the four-division structure of COBOL programs, file declarations and file organization descriptions, data type (PIC) descriptions, and structured programming in the procedure division. The COBOL format and various statements are explained in the context of these four examples.

The computing environment for this unit is COBOL on the Alpha mainframe computer. Students use PowerTerm, a terminal emulator for Windows, to edit and run COBOL programs.

**Unit 2: Object-oriented Approach and C++**

In order to meet the challenge of object-oriented methods in the computer world based on long-term considerations, we recognize that general knowledge of C++ will be an asset for IS students. The focus of this unit is placed on the difference between the traditionally structured approach and the object-oriented approach. Similarly to teaching COBOL, the features of essential notations and statements of the language, including data type, arithmetic operations, for-loop, if statement, and print statement are explained using examples. Typical examples for this unit range from a single-class program (e.g., print a flyer for a store) and two-class program (e.g., order processing) to multiple-class program (e.g., payroll processing).

C++ is an extension of C. In this unit, students actually learn both C and C++ languages. Students compare the structured approach and object-oriented approach by learning two different themes in C and C++ programming. In the functional theme of C, students learn the structure of elementary function modules and the connections between the function modules through function calling. The principle of programming in this theme is functional decomposition. In the object-oriented theme of C++, students

learn the concept and structure of classes, the characteristics of inheritance, and message sending in object-oriented programming.

We use 24% class time for this unit. The computing environment for this unit is Visual C++ of Microsoft Visual Studio 6.0.

**Unit 3: Web Pages, HTML, JavaScript, and Java**

The development of Web pages is probably the most interesting topic for business IS students. As future IS professionals, business IS students have to develop skills of computer applications in the Internet environment. These skills are acquired only when students understand client-server computing through learning HTML, JavaScript, and Java applets.

In this unit, students are required to learn the significant features and components of the integration of HTML, JavaScript, and Java applets using 26% of the entire class time. These components include essential tags of HTML, the structure of simple Web pages with JavaScript, Web pages with Java applets, and typical Java applets for animation with audio presentations. The course also gives introductions to the stand alone Java programming technique (AWT-based and non-AWT-based) as an optional part. The emphasis of this optional part is placed on a comparison of Java and C++.

In teaching this unit, typical examples of Web pages with cookie, FORM verification, and multimedia presentation supported by Java applets are presented. The requirement of this unit is the integration of HTML, JavaScript, and Java applets. The computing environment for this unit is Netscape or Microsoft Internet Explorer and Visual J++ of Microsoft Visual Studio 6.0.

**Unit 4: Graphical User Interface, Decision Support System, and Visual Basic**

In this stage of the course, students have developed basic skills in business data processing and computing on the Internet. This unit provides another important aspect of business computing: graphical user interfaces (GUI) and decision support systems (DSS).

Visual Basic is one of the popular programming languages for GUI. Students learn the major tools of developing GUI by mastering control elements including form, command button, label, text, combo menu, and program module. In the programming part, the concepts of do-loop, if statement, arithmetic operations, print and format functions are re-explained in the Visual Basic syntax by an example of on-line invoicing system. To develop their self-learning ability, students are required to use on-line help to learn built-in functions (such as NPV calculation for finance).

After learning Visual Basic, students learn Visual Basic for Applications in this unit. The design of this part is based on the considerations that business IS students must know the concept of DSS and the interaction between GUI and the model management.

A general overview of DSS is presented in this part. This is followed by a demonstration of a mini-prototype of a DSS developed in Excel Visual Basic for Applications. In this DSS example, models are the three worksheets containing variables and parameters as well as formulas. One worksheet for the Production Division calculates the production cost of a product based on material costs, labor costs, and overhead, which are all nonlinear functions of production of products. The second worksheet for the Marketing Division calculates the marketing cost of the product based on advertising costs, salesman

wages, and marketing overhead, which are also nonlinear functions of production of products. The third worksheet generates an accounting statement, which determines the net income for the company based on the sale price of the product and the total cost of production and marketing. The DSS assists the planning manager in deciding an "optimal" production level through "what-if" trials by using the designed GUI. Through this example of Visual Basic for Applications, students learn how to integrate the GUI and the model management system to develop DSS.

The computing environment for this unit is Visual Basic 6.0 of Microsoft Visual Studio 6.0 and Microsoft Excel 97.

**Examinations**

After each of the four units, students are required to write an exam. Students are presented with uncompleted computer programs with numerous blanks. Students are supposed to complete the programs, write the expected execution results of the programs, and explain the purposes of these programs. The designs of the tests strongly suggest that the global aspects of applications, including context and outcomes, are more important than the local aspects (syntax) to learn for business IS students. A sample exam is exhibited in Appendix B.

**PROJECT MODULE**

The Project Module of the course concentrates much more on "action learning", and requires students to conduct their quasi-real-world projects. In this module, students work in teams, typically, of three people. Each group applies computer programming languages discussed in the Teaching Module for their four projects. We expect that, upon the completion of this course, students are able to write programs in these computer languages to solve simple problems related to business IS.

**Project Configuration**

After each unit presented in the Teaching Module, students form project teams to share common expectations of learning. Three stages can be outlined for each project. In the first stage, students choose project topics based on their general understanding of the computer languages and their cases. In the second stage, intensive programming activities are under way for each group's project. In the final stage, the project product is formalized in a report. Although the components of reports are significantly different from project to project, the report structure adopted in four projects is almost the same. The first part of the report describes the background of the business problem to be solved. The second part provides a technical description of the project and presents source code, module diagrams, and exemplars of program execution results. The third part is a user manual.

**Project Requirements**

Because the natures of computer programming languages are very different from one to another, there is no uniform requirement for the four projects. Nevertheless, we set criteria for each project.

For file processing and COBOL, the basic requirements are:

- Structured programs for creating master files and transaction files, and manipulating these files to generate designed print reports.

- Structured diagrams for the programs.

For C++, the basic requirements are:

- Object-oriented programs with two or more classes for data manipulations.
- Object-oriented diagrams for the programs.

For HTML, JavaScript, and Java, the basic requirement is:

- Code of HTML, JavaScript, and Java applets for Web pages, hyperlinks, FORM data captures.
- Multimedia presentations including images and animations.

For Visual Basic, the basic requirements are:

- Two or more forms of GUI with various control elements, and print report generation.
- Program modules.

Students are asked to follow the structures of the cases they learned and add their own creative components for their projects. For instance, after learning the typical payroll processing system, students are generally able to write COBOL programs for their own projects such as inventory processing, sales commission processing, student credits processing, etc. Based on our observations, we are convinced that students are able to write complete programs from scratch in each of the languages for typical simple business IS problems. In our view, computer literary for IS students means "be knowledgeable with various computer languages", and such an aptitude can be developed only through hands-on practices. Appendix C exhibits a part of the course syllabus that describes the requirements of the four projects.

### **FINDINGS AND CONCLUSION**

This innovated course has been offered for three years in the College of Business at University of Massachusetts Dartmouth as a required BIS major course. Two hundred twenty three (223) students have completed this course during the past three years. After the course, each student was asked to answer the standard course evaluation questionnaire used at the university. Each question is rated on a five-point scale for students to answer. A four-point or higher mark for a question is considered a positive answer to the question for this study. Two hundred five (205) filled course evaluation forms from these students have been recorded. According to the course evaluations, 172 of 205 students gave a positive answer for the question of overall satisfaction with the course, indicating overall 84% of the students were satisfied with this course. 85% of the students considered that the material covered in this course was interesting and useful. 92% of the students favored the two-module approach to this type of course. Nevertheless, 8% students felt that the course workload is too high. According to our observations on the test results and projects, students who are eligible to take this course have no difficulty in learning the techniques outlined in this paper and conduct projects. After the course, some students have demonstrated their ability to participate more formal projects.

This approach to teaching programming for IS students is better than the traditional one-language-one-course approach in terms of skill development and knowledge acquisition. After this course, students are able to program in these different languages to solve simple business problems of data processing and Web page development, although they may not be so proficient in a particular language. These diversified basic skills will enhance students' self-learning ability. Students after this course know more

about traditional and modern computer languages than they would do after a single-language course. Knowledge of multiple languages will certainly be beneficial for them in studying other IS courses and developing hard skills for job.

Overall, the results indicate that the course discussed in this paper is a valuable component of the curriculum for business students with the major in IS. We believe that a single compact course of multiple computer languages for business IS students is useful as well as feasible. All computing resources (hardware and software) needed for this course are commonly available at education institutions. The plain textbook is available on the Internet. Thus, it is certain that our success can be replicated in other business schools.

### REFERENCES

- IS2002 [2002], IS2002 Documents, <<http://www.is2002.org/>>.
- Malik, M. A. [2000] "On the perils of programming," *Communications of the ACM*, 43(12), 95-97.
- Palma, P. D. [2001], "Why women avoid computer science," *Communication of the ACM*, 44(6), 27-29.
- Papp, R. [1998], "Job Skills in New England,"  
<<http://www.commerce.uq.edu.au/isworld/announce/msg.21-10-1998-1.html>> [Accessed August 1, 2001].
- Sleeman, D. [1986], "The challenges of teaching computer programming," *Communication of the ACM*, 29(9), 840-841.
- Wang, Shouhong and Hai Wang [2000], *Problem Solving and Programming: Essentials of Computer Languages for Commerce*, Universal Publisher, Parkland, FL,  
<<http://www.upublish.com/books/wang.htm>> [Accessed August 1, 2001].

---

---

(Appendix is omitted in this document. See the textbook for details)

## Design and Delivery of Multiple Server-Side Computer Languages Course

(Article published in *Journal of Information Systems Education*,  
Volume 22, Number 2, September 2011, pp. 159-168.)

### Abstract

Given the emergence of service-oriented architecture, IS students need to be knowledgeable of multiple server-side computer programming languages to be able to meet the needs of the job market. This paper outlines the pedagogy of an innovative course of multiple server-side computer languages for the undergraduate IS majors. The paper discusses the rationale of why the proposed pedagogy is different from and improves the traditional methods. The paper provides a description of the approach to teaching a multiple server-side computer languages course. Based on our experiences in the past years, it is concluded that a single course of multiple server-side computer languages is useful and feasible for the IS programs.

**Keywords:** IS curriculum, server-side computer languages, pedagogy, multiple programming languages

### 1. Introduction

There have been critical discussions on the IS curriculum design during the last several years (Topi et al. 2010). The most notable trend in the IS curriculum renewal movement is to develop more new IS electives to meet the needs of the job market of IS graduates (Drinka and Yen 2008). This need in the job market has considerable implications for the designing of IS elective courses in the educating of the next generation of IS professionals. IS students must acquire the fundamental theories of IS as well as the essential practical skills of the computer applications, while developing the life-long learning ability in information technology during their IS education. (Surendra and Denton 2009; Topi et al. 2010). Technical skills should focus more on problem solving and practical applications (Downey et al. 2009). The IS curricula have changed over the past years to meet the requirements of the job market as well as the requirements of the accreditation organizations such as AACSB and ABET. Nevertheless, programming remains a core requirement in most IS programs, which is one of the most difficult subjects in the IS curriculum (Sleeman 1986; Malik 2000; Palma 2001; Robins et al. 2003; May and Dhillion 2009). This paper reports how this challenge is met by designing the contents of a course of multiple server-side computer programming languages for undergraduate IS majors.

### 2. Overview of the Course

In the modern service-oriented age, the development and maintenance of Web based applications still relies heavily on applications of third generation computer languages regardless of the advances in fourth generation computer languages and a variety of software packages. To meet the challenges of the ever-changing information technologies, educators need to offer courses of important server-side programming languages for their undergraduate IS majors. On the other hand, undergraduate IS majors cannot afford to learn multiple server-side computer languages on the one-language-one-course basis. The key to the solution to this problem is to make a pedagogical paradigm shift and to develop a course of multiple server-side computer languages.

Few guidelines for IS elective courses of server-side computer programming can be found in the literature or on the Internet. The selection of server-side languages for a course is a crucial task for the pedagogy design. The design components of such a course are based on four considerations. First, the selected server-side languages must be commonly used in the industry. Second, the selected server-side languages must be representative, and should cover essential features of all kinds of server-side languages. Third, the selected server-side languages do not require additional computing resources in our computing lab. Fourth, the total workload for students should be manageable. Taking these factors into account, three major server-side computer programming languages were selected for our course. These languages are ASP.NET (with VB.NET), PHP, and XML (with XSLT, DTD and XML Schema).

Due to time constraints, it is impossible for students to learn all these server-side languages in great detail. Nevertheless, students in this course are expected to have general knowledge of server-side languages as well as to be able to develop basic skills of programming. The central methodology applied to this course is the learning of languages through typical examples. Specifically, we teach typical problems of Web applications and their solutions through the use of these computer languages.

The course described in this paper is entitled “Web Application Development and Programming,” and is designed as an elective course for junior IS majors who are pursuing the careers as application developers or Web content managers. The prerequisite of this elective course is an introductory computer programming course, which provides an introduction to client-side computer languages. Although the contents of the introductory programming course depend upon the individual instructor, the course normally covers HTML, JavaScript, and VB.NET. The elective course described in this paper is taught over one semester, normally 3 credit hours over 14 weeks. In its design, this course consists of two distinct modules. The Teaching Module provides an overview of representative server-side computer languages in service-oriented architectures. The Project Module provides an opportunity to apply the server-side computer languages involving hands-on projects.

### **3. Justification of the Course**

The structure of the 2010 IS model curriculum (Topi et al. 2010) includes two parts: core IS courses and elective IS courses. The core IS courses include Foundations of IS; Enterprise Architecture; IS Strategy, Management, and Acquisition; Data and Information Management; Systems Analysis and Design, IT Infrastructure; and IT Project Management. The elective IS courses include Application Development; Business Process Management; Collaborative Computing; Data Mining/Business Intelligence; Enterprise Systems; Human-Computer Interaction; Information Search and Retrieval; IT Auditing and Controls; IT Security and Risk Management; Knowledge Management; and Social Informatics. The 2010 curriculum recommendation clearly states that, unlike its predecessor model curricula, it does not provide specific courses that address the IS foundational skills and knowledge or domain-specific skills and knowledge. The design and delivery of IS courses, especially electives, for an IS program depend upon the computing environment and its specific needs.

Web services, software-as-a-service, and cloud computing are all important elements in the modern service-oriented architectures of computer-based systems. The present server-side programming course provides the necessary body of knowledge for the current trends. In terms of the 2010 IS Model Curriculum (Topi et al. 2010, Figure 6), the proposed course fits “Application Development” which is on the top of the list of elective IS courses and is relevant to all career tracks. In our IS programs, e-commerce is a concentration area. Our experiences in the past several years have indicated that this course is an important component of the e-commerce concentration, and meets various career tracks which are suggested in the 2010 curriculum recommendation. During the past three years, the IS students’ demands for this elective course have been increasing significantly although the overall enrollment of IS majors remains steadily low.

### **3.1. Need for learning multiple server-side computer languages**

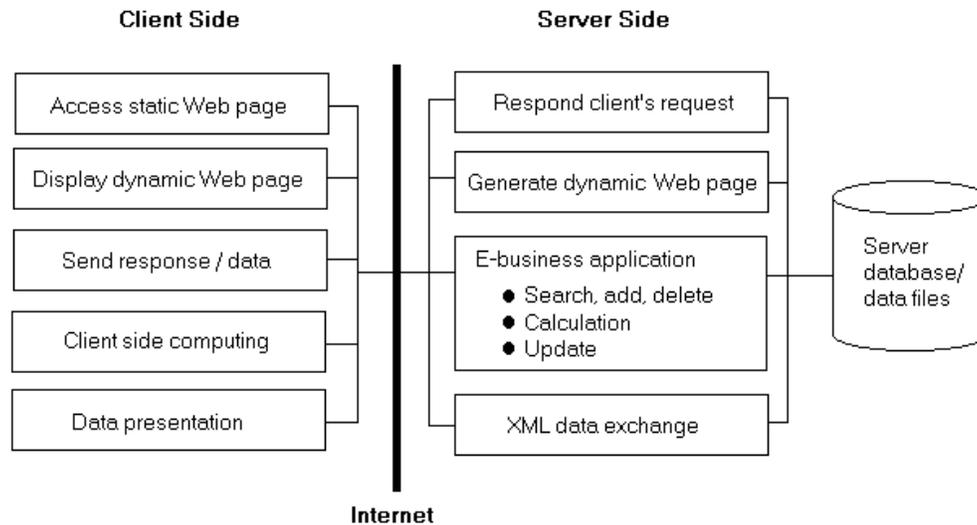
Due to historical and technical reasons, there have been many server-side computer languages. The need for learning multiple computer languages can be perceived from a variety of views. For instance, Perl is still important since many legacy Web-based systems are Perl-based. However, modern large service-oriented systems heavily rely on Java and .NET. On the other hand, PHP is widely used in many small scale service-oriented applications. Our argument is that, given the variety of client-server middleware in the computer-based systems, undergraduate IS majors need to learn the distinct features of the major server-side languages.

### **3.2. Sufficient coverage of important concepts of server-side programming**

The course shall cover the key concepts of server-side computing that supports interaction between the clients and server as well as the actions on the server. The key concepts that are particularly relevant to service-oriented architectures are identified and are summarized as follows.

#### *3.2.1. Job division in service-oriented architectures*

The architecture of web-based applications covers a broad range of subjects, including information orientation, communication orientation, and service orientation. In the present study, the focal point is placed on the service-oriented architecture. Job division between the client and server in the service-oriented architectures is an essential concept for undergraduate IS majors. In the service-oriented architectures, the client side is responsible for data presentation and data input, and the server side is responsible for data processing, as illustrated in Figure 1.



**Figure 1. Client-Server Job Division in Service-Oriented Architectures**

### 3.2.2. *The mechanism of service-oriented process*

Service orientation is implemented through interactions between the client and the server. Client-side computing programs, Web browser, the Internet protocol, server-side programs, and server database or data files comprise the chain of service-oriented processes. The ultimate service-oriented processes on the server side are recording, updating, and searching of data on the server in response to the requests from the client side.

### 3.2.3. *Dynamic Web pages generated by server-side programs*

One of the key tasks of server-side programs is the generation of dynamic Web pages, no matter what server-side computer language is used to implement service-orientation. Dynamic Web pages implement contingent actions of the server in response to the client's specific requests.

### 3.2.4. *Uniform format of data exchange*

XML, the uniform data format for databases, has become a necessary component in service-oriented architectures. XML and its companion languages (XSLT, XML Schema, etc.) make data accurate in exchanging, efficient in searching, and independent of presentation.

In summary of the above discussion, there is little doubt that knowledge of various server-side computer languages are substantial, and can contribute to the career development for undergraduate IS majors. Now, the question is how we can teach these multiple server-side languages in a feasible way.

## 3.3. Feasibility of teaching multiple server-side languages

It is impossible to teach multiple languages by using the traditional one-language-one-course approach. To compact material of multiple languages into a single course, we must change the traditional approach to teaching computer languages that begins with syntax and ends with disjointed examples of algorithms. Typically, we use simple examples of client-server end-to-end interaction to teach such a course. We take advantage of synonyms; that is, despite the variety of syntax in the different languages,

languages share many common key concepts. Once a key concept is introduced in one language, it should be understandable for students when learning another language.

### **3.4. The course adds more value to student learning**

This innovative course is fast-paced, and encourages self-learning on the students' part. Since the nature of this course is practice-oriented, multiple small-scale projects will be required. The central point of the shift from one-language-one-course to multiple-languages-one-course is to add more value to student learning by giving students discipline in self-learning and encouraging them to apply learned knowledge to the real world. According to our observations, students feel stress in the same way as in any other programming course; but, the never felt bored in this course.

## **4. Teaching Module**

It was difficult to find a single integrated textbook that meets our course needs. To give students a comprehensive guideline for their studies, lecture notes have been developed for this course during the past years. These lecture notes have been revised many times, and recently have been included in a textbook of computer programming languages for IS students (reference is available upon request).

The Teaching Module is divided into three units based on the selected server-side languages. Each unit has a written exam. A sample course syllabus is exhibited in Appendix I. The features of the course competency, relevant paradigms, and service-oriented applications are presented below.

### **4.1. Unit 1: ASP.NET**

Compared with the Java platform, ASP.NET may not be as efficient as Java in terms of execution performance, but is a top popular server-side language due to the popularity of the Windows platform. In our experience of teaching server-side languages, ASP.NET is particularly superior in a computing environment in providing a seamless connection with databases (MS SQL Server or Access).

An ASP.NET program has two parts: the server processing logic and the Web form. We use VB.NET for the part of user interface logic. Although the prerequisite course of this course includes VB.NET, we still use the first two or three classes for the reviewing VB.NET.

The major components of this unit include the structure of ASP.NET program, HTML controls and Web controls, code-behind programming framework, and reading and writing database or files on the server.

A great feature of ASP.NET is the seamless connection with Access databases through ADO.NET. One interesting part of this unit is the use of SQL for Web applications. Although most students in this course have not taken a database course, they seem to be able to deal with simple SQL embedded in ASP.NET. Typical client-server end-to-end examples for this unit include online order processing and student database searching.

We use 40% of the class time for this unit. Our computing lab is equipped with PCs using the Windows platform. The computing environment for this unit is Microsoft Visual Studio 2010.

### **4.2. Unit 2: PHP**

To meet the challenge of information technology in the service-oriented world based on long-term considerations, we recognize that general knowledge of PHP will be an asset for undergraduate IS majors since PHP is free open source software and is widely used in service-oriented applications. In terms of syntax, PHP is very similar to Perl which is considered to be old but is still running on many servers.

The focus of this unit is placed on the structural difference between ASP.NET and PHP. Unlike ASP.NET, PHP does not have a division in the server processing logic and the Web form; instead, a PHP program mixes both server-side data processing and printing (or generating) a dynamic Web page for the client.

Similar to teaching ASP.NET, the major features of PHP, including commonly used commands, reading/writing data files on the server, and data relay through multiple forms, are explained using examples. Typical client-server end-to-end examples for this unit include online order processing and airticket reservation.

The computing environment for this unit is EasyPHP (2011), an open source software PHP environment. EasyPHP is easy to install and easy to use. Students take the responsibility to install EasyPHP on their computers. PHP with MySQL is a good combination for delivering the concepts of server-side programming. We use 25% of the class time for this unit. Our observations have shown that, in comparison with ASP.NET, PHP has less built-in features and is easier to learn when students have learned ASP.NET.

Currently, we are using the Windows platform for the PHP unit. It will certainly be a new direction for this unit if in the future we are going to integrate PHP and databases on Linux servers which are widely used in industry client-server systems.

#### **4.3. Unit 3: XML**

XML has been widely applied in Web applications, and the application is an important body of knowledge for undergraduate IS majors. Although the concept of XML is not difficult to learn, students often have difficulties in this unit. The problems with this unit are not with XML itself, but with the confusion caused by so many XML companion languages.

The major components of this unit include features of XML instance documents, data structure of an XML document, Cascading Style Sheets (CSS), Extensible Style Language (XSL), Document Type Definition (DTD), XML Schemas, and XML document validation. Typical client-server end-to-end examples for this unit are the use of a single data set in the XML format for different purposes in separate service-oriented applications.

This unit includes XHTML and a very brief overview of XBRL. However, these components are optional.

For the time being, there is little direct connection between XML and ASP.NET or PHP. It would be interesting to watch closely how the IT industry establishes application linkages between XML and ASP.NET or PHP.

We use 35% of the class time for this unit. The computing environment for this unit is Microsoft Internet Explorer. Free user-friendly XML validators such as CoreFiling (2011) are also used for this unit.

#### **4.4. Examinations**

After each of the three units, students are required to take an exam. Students are presented with uncompleted computer programs with numerous blanks. Students are required to complete the programs, explain the purposes of these programs, and sketch the expected execution results of the programs. The designs of the tests strongly suggest that the global aspects of service-oriented applications, including context, client-server interaction, data processing on the server, and service-oriented process outcomes, are more important than the local aspects (syntax) to learn for undergraduate IS majors.

### **5. Project Module**

The Project Module of the course concentrates much more on "action learning", and requires students to conduct their quasi-real-world projects. In this module, students were encouraged to work in small teams, typically, of two people. Nevertheless, many students chose to work on projects without joining team. Each group (or individual) applies the server-side languages discussed in the Teaching Module for their three projects. It is expected that, upon the completion of this course, these students are able to write programs in these computer languages to solve simple problems related to service-oriented applications.

### **5.1. Project Configuration**

After each unit presented in the Teaching Module, students form project teams if they choose to do so to share common expectations of learning. Three stages can be outlined for each project. In the first stage, students choose project topics based on their general understanding of the server-side languages and their cases. In the second stage, intensive programming activities are undertaken for each project. In the final stage, the project product is formalized in a report. Although the components of reports may differ significantly from project to project, the report structures adopted in three projects are almost the same. The first part of the report will describe the background of the business problem to be solved. The second part of the report provides a technical description of the project including source code. The third part of the report provides screenshot examples of program execution results.

### **5.2. Project Requirements**

We clearly set criteria for each project for assessment, as briefly described below.

#### *5.2.1. Minimum requirements for ASP.NET projects*

Possible project topics include online shopping, online survey, online billing, online payment, etc. The minimum requirements for ASP.NET include: An HTML home page that starts an ASP.NET program, followed by two times of interaction between the client and server implemented by ASP.NET; At least 1 data file (.txt file) and 1 database table (MS Access) used for the programs for data storage and search.

This course emphasizes on the interaction between client and server, but de-emphasizes the static Web page itself (such as hyperlinks, client-side calculations and image manipulations through JavaScript).

A project report includes:

- Description of the project;
- Source code of the home page and all programs;
- The screenshots that illustrate the interaction between client and server; and
- Data files and MS Access database that will illustrate the data storage on the server, and sample data for the project.

#### *5.2.2. Minimum requirements for PHP projects*

The requirements for PHP projects are similar to the requirements for ASP.NET, except for the database which is not required for PHP, because the setting database connections for PHP is beyond the scope of the server-side language course. Nevertheless, many students were able to set MySQL or SQL Server for their projects by themselves through learning from the Internet.

### 5.2.3. *Minimum requirements for XML projects*

The minimum requirements for XML projects are one XML document, its Schema and validation, and two XSLT programs for the XML document for two distinct applications.

A project report should include:

- Description of the project;
- Source code of all programs (XML, XSLT, Schema);
- Tree diagram for the XML document;
- Validation of the XML document against its schema;
- The screen shots of the two distinct applications.

Students are asked to follow the structures of the service-oriented application examples they learned in the Teaching Module and to add their own creative components for their projects. For instance, after learning the typical online ordering process, students are generally able to write ASP.NET and PHP programs for their own projects such as hotel reservation, online auction, etc. Based on our observations, we are convinced that students are able to write complete programs from scratch in each of the languages we taught for typical simple service-oriented application projects. Computer literacy for undergraduate IS majors means "be knowledgeable with a variety of computer languages", and such an aptitude can be developed only through hands-on practices. (Woszczyński et al. 2005; Kung et al. 2006).

## 6. Findings and Conclusion

This server-side language course has been offered in our IS programs at the two universities of the co-authors during the past six years. Although the teaching evaluation questions at the two universities were not same, the format and major aspects were similar and comparable. Only the questions of overall satisfaction with the course and free-format comments were considered and analyzed for this study. Questions were rated on a five-point scale for students to answer. A four-point or higher mark for a question was considered a positive answer to the question. Eighty seven (87) filled course evaluation forms from this course have been recorded for this study. According to the course evaluations, all students gave a positive answer for the question of overall satisfaction with the course. According to our observations of the exam results and projects as well as the comments from the students, students who were eligible to take this course have no difficulty in learning the techniques outlined in this paper and conducting projects. After the course, some students have demonstrated their ability to participate more formal service-oriented application projects. We have compelling examples of follow up comments by those who took this course. In fact, we have collected several comments from the students who are working in the Web application development field subsequent to taking this course, and have presented them to our current students (see Appendix II). Although our sample size was relatively small, we are convinced that knowledge of multiple server-side languages will certainly be beneficial for the IS programs.

Overall, the results indicate that the course discussed in this paper is a valuable component of the renewal curriculum for IS programs. We believe that a single compact course of multiple server-side languages for undergraduate IS majors is useful as well as feasible. All computing resources (hardware and software) needed for the implementation of this course are commonly available at all educational institutions. Thus, it is certain that our success can be replicated in other IS programs.

One of the most important aspects of effective IS education is to help IS students to develop problem solving skills to meet the challenges of the fast changing IS field. IS educators need to have a greater understanding of problem solving schemes in order to design innovative curricula that will emphasize students' practical skill sets. Our study has made contributions to innovative education as it has initiated and implemented a course of three major server-side programming languages for the IS majors. The summarized pedagogy of this course can be applied by others to change the traditional methods of teaching computer programming languages by shifting from single-emphasis to variety-emphasis. The most compelling implication of this study for the IS education is the recognizing of the potentially positive effect of pedagogical re-design in enhancing the students' problem solving skill sets.

### References

- CoreFiling. 2011. XML Schema Validator <<http://tools.decisionsoft.com/schemaValidate/>>, [accessed August 3, 2011].
- Downey, J. P., McGaughey, R., and Roach, D. 2009. "MIS versus computer science: an empirical comparison of the influences on the students' choice for major," *Journal of Information Systems Education*, 20(3), 357-368.
- Drinka, D. and Yen, M. Y. 2008. "Controlling curriculum redesign with a process improvement model," *Journal of Information Systems Education*, 19(3), 331-342.
- EasyPHP. 2011. Open source software PHP <<http://www.easyphp.org/>>, [accessed August 3, 2011].
- Guthrie, C. 2010. "Towards greater learner control: Web supported project-based learning," *Journal of Information Systems Education*, 21(1), 121-130.
- Kung, M., Yang, S. C., and Zhang, Y. 2006. "The changing information systems (IS) curriculum: a survey of undergraduate programs in the United States," *Journal of Education for Business*, 81(6), 291-299.
- Malik, M. A. 2000. "On the perils of programming," *Communications of the ACM*, 43(12), 95-97.
- May, J., and Dhillon, G. 2009. "Interpreting beyond syntactics: a semiotic learning model for computer programming languages," *Journal of Information Systems Education*, 20(4), 431-438.
- Palma, P. D. 2001, "Why women avoid computer science," *Communication of the ACM*, 44(6), 27-29.
- Robins, A., Rountree, J., and Rountree, N. 2003. "Learning and teaching programming: a review and discussion," *Computer Science Education*, 13(2), 137-172.
- Sleeman, D. 1986. "The challenges of teaching computer programming," *Communication of the ACM*, 29(9), 840-841.
- Surendra, N. C., and Denton, J. W. 2009. "Designing IS curricula for practical relevance: applying baseball's 'Moneyball' theory," *Journal of Information Systems Education*, 20(1), 77-85.
- Topi, H., Valacich, J. S., Wright, R. T., Kaiser, K., Nunamaker, J. F. Jr., Sipior, J. C., and Vreede, G., J. d. (2010). "IS 2010: Curriculum guidelines for undergraduate degree programs in information systems," *Communications of the Association for Information Systems*, 26, Article 18.
- Woszczynski, A. B., Guthrie, T. C., and Shade, S. 2005. "Personality and programming," *Journal of Information Systems Education*, 16(3), 293-299.

**(\*\* Appendix A is omitted in this documents)**

---

**Appendix II. Samples of Students' Feedback Comments on the Course**Former Student #1:

Hello, Professor:

I think it was a great help in getting my current job as a Web Developer. I recently finished building <http://www.xxxx.com/> by myself using drupal. My next project is to rebuild the entire software suite that they sell to local municipalities, etc. This software suite is enormous in scale so it is a rather scary thing knowing what is ahead of me. I will be building the software website using visual studio 2008, asp.net(vb.net), MS SQL Server 2008, and IIS 7.5. One of the things that I noticed at work and during my job searching is that mostly everyone today is using SQL Server or some other database software and visual studio. My company currently uses Access and SQL Server depending on which release the customer has. So I am gaining experience in both platforms. My main reason for touching base with you was just to thank you for everything but also to just let me know that it might be a good thing for your future students to learn some SQL Server and MySQL. Mainly it is important to be used to the Visual Studio environment. That has been the biggest adjustment I have had to make since joining this company. Learning Visual Studio and database interaction with SQL Server has been by far the most difficult part of the job. Again, thank you for everything you taught me, Professor.

A. D.

Former Student #2:

Hi, Professor:

As an Instructional Technologist, my tasks include testing, supporting and extending functionality to a variety of open source and commercial applications related to teaching and learning. In addition, I often must work with data and content encoded to meet industry standards. Exposure and understanding gained through MIS 312 impacts all of these functions. This is due to PHP, ASP .NET and XML being among the most prevalent languages used in industry today. Prior to my experience in MIS 312, I had not yet been exposed to ASP .NET programming. The experience gained through the ASP project enables me to better support applications currently being evaluated for deployment in this organization. XML is a major component in ensuring content interoperability for many modern learning management and media delivery platforms that I work with and evaluate. Because of the knowledge gained in MIS 312, I now have a much greater understanding of XML schema development. Even though I do work with PHP frequently, the freedom of the course project allowed me to experiment with relevant functionality I had not yet had the opportunity to work with professionally.

D. G.

Former Student #3:

Dear Professor,

I believe MIS 312 has greatly impacted my job hunting and work after the MIS program. I am currently a Programmer/Analyst for XXX Information Technology implementing Healthcare Information Systems for the XXX applications. Although XXX uses its own proprietary languages for its applications, it does use some of the web based technologies. For example, I was recently asked to write a web based version (HTML & XML) of a document for my group due to the fact that I have experience with those technologies. The experience I speak of is the experience I gained in the MIS 312 class. The class provided me with the skills needed to complete these types of tasks with ease and efficiency.

It is my opinion, through my experience, that these technologies are still being utilized. When hunting for technical positions, being well rounded in these technologies is an absolute plus. Even if it is not utilized at a particular company or organization I believe knowing the technologies shows you ability to understand the connection between client and server. It also shows you ability to learn new languages quickly and to be able to apply them for business purposes. Numerous positions I researched in my job hunting required experience in ASP.NET, PHP and XML.

Thanks,

R. D.

## 2. General Teaching Strategies

Teaching strategies are guidelines and plans applied to the teaching activities to improve classroom practice and enhance student learning. Here, several general teaching strategies specifically for these programming courses are suggested based on the authors' experiences in teaching this course.

### *Teach Programming by Emphasizing the Structures of the Programs and Interconnections of the Instructions*

As explained in the Introduction of the textbook, instead of emphasizing the syntax of individual commands, we emphasize the structures of the programs and interconnections of the instructions. We analyze typical examples for each language, and use fill-blank questions (self-exercise questions after each chapter) to re-enforce students' learning. We ask students to find those answers by going through the programs and tracing the source for the answers. For instance, for Chapter 2, Self-Exercise, Question (11), students should be able to fill the blank in line 8 using the information in line 20, as these two lines are interconnected.

Tests for programming are open-book, no computer. Each test for a language basically consists two parts: (1) filling blanks and (2) sketching the expected execution result. We have found that these styles of tests are effective for testing students' understanding of the program structures and interconnections of the instructions.

### *Hands-On and Hands-on Again*

Hands-on is the central approach to letting students fully understand programming. Students are encouraged to make changes to the original textbook examples for their own purposes. The more changes a student makes, the higher grade he/she will receive for the project.

### *Make the Instructional Module and Project Module Cohesive*

An instructional module usually carries on for the entire semester, while project modules start weeks later after students learn the context. The instructor shall help students to balance the workload across the course by specifying the agenda in the syllabus clearly. More importantly, the instructor shall make the material of the multiple languages cohesive, and connect the basic concepts of each language taught through class discussion.

*Maintain Continuous Progress and Monitor Self-Paced Learning*

Milestones are needed to check the progress of assignments and projects. The instructor shall continuously offer suggestions to individual groups. Self-paced learning approach is appropriate for the programming courses since provision of classroom tutorials is insufficient in many cases. Instructions for self-paced learning are then necessary. This teaching strategy helps to build bond between the instructor and students, and provides a mechanism of quality control for the course projects.

*Effective Cooperative Learning*

Cooperative learning is the use of small groups so that students work together to maximize their own and each other's learning. The cooperative learning strategy is commonly used in business education. However, cooperative learning does not guarantee positive learning experiences. The instructor shall encourage students to use peer-reviews to maintain the quality of cooperative learning.

### 3. Lab Preparation

Teaching programming courses in business schools would be much more difficult than in computer science schools because the lab conditions in business schools are usually poor. Instructors of programming language courses know best on their extra duties compared with other courses. They not only prepare lectures, hands-on topics, but, more importantly, spend much time to prepare the lab before even the semester hasn't started. If the lab is supportive, the job of the instructor is easier. If the lab is problematic, the instructor could be in trouble. More importantly, students suffer. The following is a guideline for lab preparation and an alternative to computing lab.

*(1) Select the computing environment for each language you are going to teach.*

The computing environments of the computer languages covered by the textbook are mostly free or those your university has licensed.

For C++, the textbook shows the use of Microsoft Visual Studio for C++. However, you may choose to use Dev-C++, an open source C++ environment.

For HTML, JavaScript, CSS and XML, you virtually do not need any thing more than Notepad and IE Explorer or Netscape which are usually available in any labs for business schools.

For PHP, you can download freeware. You can either request the computer service center to install it on the server, or ask students to install it on their own laptops.

For VB.NET, C#.NET, ASP.NET, and SQL, you need a Microsoft Visual Studio .NET suite. If students have them on the laptops, server installation is unnecessary.

*(2) Set the environment for the language.*

If you use your lab for the language you are going to teach, you need to make a request for your computer service center following the university/college procedure. At most universities you need to give the computer service center at least a month before you can use the environment. If you use freeware, more likely you need to submit a document of the license agreement and a master CD with your applications.

*(3) Test the environment.*

After the installation of the computing environment for the language, you need to test the environment. If a "Hello, World" example does not work, the environment does not function.

You must find out the problems. Generally speaking, if the “Hello, World” example works, the environment is considered to be set correctly. Needless to say, before you test all programs listed in the textbook, there always are potential problems with the different versions of the compiler or interpreter.

*(4) An alternative to computing lab.*

An alternative to computing lab is to let MIS majors use their own laptop, and use commonly available Microsoft environment and/or download free compilers and interpreters for your course. As explained by the textbook appendices, students can install their personal server for server-side programming languages Java servlets and PHP. ASP.NET programming can run on PC in the Microsoft Visual Studio environment. The advantage of this approach is that students can learn more about the environment, and the risk of lab malfunction can be reduced. The disadvantage of this approach is that the job of the instructor might be actually more difficult since she/he must look after the installation for every student. To the authors’ experiences, this alternative might not be suitable of the introductory course, but can be good for the advanced programming course.